

DRAFT

NIST CYBERSECURITY PRACTICE GUIDE

DOMAIN NAME SYSTEMS-BASED ELECTRONIC MAIL SECURITY

Scott Rose

William C. Barker

Santos Jha

Chinedum Irrechukwu

Karen Waltermire

NIST SPECIAL PUBLICATION 1800-6

(INCLUDING PARTS A, B, C)

ADDITIONAL CONTENT:

https://nccoe.nist.gov/projects/building_blocks/secured_email



DOMAIN NAME SYSTEMS-BASED ELECTRONIC MAIL SECURITY

1800-6A**Executive Summary****1800-6B****Approach, Architecture, and Security Characteristics***For CIOs, CSOs, and Security Managers***1800-6C****How-To Guides***For Security Engineers*

Scott Rose

Information Technology Laboratory
National Institute of Standards and Technology

William C. Barker

Dakota Consulting
Silver Spring, MD

Santos Jha

Chinedum Irrechukwu
The MITRE Corporation
McLean, VA

Karen Waltermire

National Cybersecurity Center of Excellence
National Institute of Standards and Technology

November 2016

U.S. Department of Commerce
Penny Pritzker, SecretaryNational Institute of Standards and Technology
Willie May, Under Secretary of Commerce for Standards and Technology and Director

DISCLAIMER

Certain commercial entities, equipment, products, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by NIST or NCCoE, nor is it intended to imply that the entities, equipment, products, or materials are necessarily the best available for the purpose.

National Institute of Standards and Technology Special Publication 1800-6
Natl Inst. Stand. Technol. Spec. Publ. 1800-6, 221 pages (November 2016)
CODEN: NSPUE2

Organizations are encouraged to review all draft publications during public comment periods and provide feedback. All publications from NIST's National Cybersecurity Center of Excellence are available at <http://nccoe.nist.gov>.

For additional information and supplemental materials relating to this document and project, please see https://nccoe.nist.gov/projects/building_blocks/secured_email.

Comments on this publication may be submitted to: dns-email-nccoe@nist.gov

Public comment period: November 2, 2016 through December 19, 2016

National Cybersecurity Center of Excellence
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, MD 20899
Mailstop 2002
Email: dns-email-nccoe@nist.gov

NATIONAL CYBERSECURITY CENTER OF EXCELLENCE

The National Cybersecurity Center of Excellence (NCCoE) at the National Institute of Standards and Technology (NIST) addresses businesses' most pressing cybersecurity problems with practical, standards-based solutions using commercially available technologies. The NCCoE collaborates with industry, academic, and government experts to build modular, open, end-to-end reference designs that are broadly applicable and repeatable. The center's work results in publicly available NIST Cybersecurity Practice Guides, Special Publication Series 1800, that provide users with the materials lists, configuration files, and other information they need to adopt a similar approach.

To learn more about the NCCoE, visit <http://nccoe.nist.gov>. To learn more about NIST, visit <http://www.nist.gov>.

NIST CYBERSECURITY PRACTICE GUIDES

NIST Cybersecurity Practice Guides (Special Publication Series 1800) target specific cybersecurity challenges in the public and private sectors. They are practical, user-friendly guides that facilitate the adoption of standards-based approaches to cybersecurity. They show members of the information security community how to implement example solutions that help them align more easily with relevant standards and best practices.

The documents in this series describe example implementations of cybersecurity practices that businesses and other organizations may voluntarily adopt. The documents in this series do not describe regulations or mandatory practices, nor do they carry statutory authority.

DRAFT

Domain Name Systems-Based Electronic Mail Security

³ Executive Summary

- ⁴ ■ Both public and private sector business operations are heavily reliant on electronic mail (email)
⁵ exchanges but the integrity of these transactions is often at risk, including financial and other
⁶ proprietary information as well as the privacy of employees and clients.
- ⁷ ■ Protocols such as Transport Layer Security (TLS), Secure/Multipurpose Internet Mail Extensions (S/
⁸ MIME), Domain Name System Security Extensions (DNSSEC), and Domain Name System (DNS)
⁹ Authentication of Named Entities (DANE) exist and are capable of providing needed email security
¹⁰ and privacy protection.
- ¹¹ ■ Impediments such as the absence of comprehensive configuration instructions for a composed set of
¹² mail client, mail transfer agents, and DNS security components, absence of resource guides to easily
¹³ implemented software libraries and software applications for system administrators, and functional
¹⁴ characteristics of security applications that negatively impact the performance of email systems have
¹⁵ limited adoption of these existing security and privacy protocols.
- ¹⁶ ■ Operating email systems without employing available security and privacy protocols increases the
¹⁷ opportunities for attackers to breach sensitive enterprise information by introducing false addresses
¹⁸ into mail messages, disrupting secure communication signaling, and improving the probability of
¹⁹ successfully inducing enterprise users to open malicious attachments - still the most common method
²⁰ for introducing malware and breaching enterprise systems.
- ²¹ ■ The National Cybersecurity Center of Excellence (NCCoE) developed a set of example DNS-based
²² email security solutions that organizations can use to facilitate implementation of security and privacy
²³ protocols, thus reducing the likelihood of a data breach. The solution sets include tools that support
²⁴ installation and set-up of trustworthy email systems.
- ²⁵ ■ The security characteristics in this guide are informed by guidance and best practices from standards
²⁶ organizations. How the solution set addresses security requirements and best practices is addressed
²⁷ in a volume that includes the security approach, architecture, and security characteristics.
- ²⁸ ■ The NCCoE's approach uses both open source and commercially available products that can be
²⁹ included alongside current mail products in existing infrastructure.
- ³⁰ ■ The example solution is described in a "How To" guide that shows how to implement a set of
³¹ standards-based, commercially available cybersecurity technologies in the real world. The guide will
³² help organizations utilize technologies to reduce the risk of untrustworthy email, while saving them
³³ research and proof of concept costs.

34 THE CHALLENGE

35 Whether the security service desired is authentication of the source of an email message or assurance
 36 that the message has not been altered by or disclosed to an unauthorized party, organizations must
 37 employ some cryptographic protection mechanism. Economies of scale and a need for uniform security
 38 implementation drive most enterprises to rely on mail servers and/or Internet service providers (ISPs) to
 39 provide security to all members of an enterprise. Many current server-based email security mechanisms
 40 are vulnerable to, and have been defeated by, attacks on the integrity of the cryptographic
 41 implementations on which they depend. The consequences of these vulnerabilities frequently involve
 42 unauthorized parties being able to read or modify supposedly secure information, or to use email as a
 43 vector for inserting malware into the system in order to gain access to enterprise systems or information.
 44 Protocols exist that are capable of providing needed email security and privacy, but impediments such as
 45 unavailability of easily implemented software libraries and software applications characteristics that
 46 complicate operation of email systems have limited adoption of existing security and privacy protocols.

47 THE SOLUTION

48 The Domain Name System-Based Security for Electronic Mail (Email) project has produced a proof of
 49 concept security platform that demonstrates trustworthy email exchanges across organizational
 50 boundaries. The goals of the project include authentication of mail servers, signing and encryption of
 51 email, and binding cryptographic key certificates to the servers. The Domain Name System Security
 52 Extension (DNSSEC) protocol is used to authenticate server addresses and certificates used for Transport
 53 Layer Security (TLS) to DNS names. The business value of the security platform demonstrated by this
 54 project includes improved privacy and security protection for users' operations and improved support for
 55 implementation and use of the protection technologies. The platform also expands the set of available
 56 DNS security applications and encourages wider implementation of DNSSEC, TLS and S/MIME to protect
 57 internet communications.

58 Project deliverables include:

- 59 ■ demonstration prototypes of DNS-based secure email platforms
- 60 ■ this publicly available NIST Cybersecurity Practice Guide that explains how to employ the platform(s)
 61 to meet industry security and privacy best practices as well as requirements for federal government
 62 agencies
- 63 ■ platform documentation necessary to efficiently compose a DNS-based email security platform from
 64 off-the-shelf components
- 65 ■ recommendations for effective implementation in a manner that is consistent with applicable
 66 standards documentation

67 Approach

68 The secure email project involves composition of a variety of components that have been provided by a
 69 number of different technology providers, including Microsoft Corporation, the Internet Systems
 70 Consortium, Secure64, Fraunhofer IAO, and Stichting NLnet Laboratories. Each of these collaborators has
 71 entered into a Cooperative Research and Development Agreement (CRADA) with NIST to participate in
 72 this consortium effort. These components include client systems, DNS/DNSSEC services, mail transfer
 73 agents (MTA), and certificate sources.

74 We demonstrate how security can be supported through standards-based configuration and operation
 75 DNS servers, electronic mail applications and MTAs in a manner that supports trustworthy email by the
 76 organization.

77 The guide:

- 78 ■ identifies the security characteristics needed to sufficiently reduce the risks to information exchanged
- 79 by email
- 80 ■ maps security characteristics to standards and best practices from NIST and other organizations
- 81 ■ describes a detailed example solution, along with instructions for implementers and security
- 82 engineers on efficiently installing, configuring, and integrating the solution into existing IT
- 83 infrastructures
- 84 ■ provides an example solution that is operationally practical and evaluates the performance of the
- 85 solution in real-world scenarios

86 **BENEFITS**

- 87 Our example solution has several benefits, including the following:
- 88 ■ reduces risk so that employees are able to exchange personal and enterprise information via email
- 89 with significantly reduced risk of disclosure or compromise
- 90 ■ enables the use of existing security protocols more efficiently and with minimal impact to email
- 91 service performance
- 92 ■ integrates capabilities into various server and client IT infrastructure environments
- 93 ■ enhances visibility for system administrators into email security events, providing for recognition of
- 94 authentication failures that could result in device and data compromises
- 95 ■ implements both commercial and open source industry standard network and email security controls
- 96 reducing long term costs and decreasing the risk of vendor lock-in
- 97 ■ can be extended to other enterprise information exchange technologies that are growing in use (e.g.,
- 98 text messages, chat)

99 **TECHNOLOGY PARTNERS AND COLLABORATORS**

100 The technology vendors who participated in this project submitted their capabilities in response to a call

101 in the Federal Register. Companies with relevant products were invited to sign a Cooperative Research and

102 Development Agreement with NIST, allowing them to participate in a consortium to build this example

103 solution. We worked with:

- 104 ■ Microsoft Corporation
- 105 ■ NLnet Laboratories
- 106 ■ Secure64
- 107 ■ Internet Systems Consortium
- 108 ■ Fraunhofer IAO

109 **SHARE YOUR FEEDBACK**

110 You can get the guide through the NCCoE web site, <http://nccoe.nist.gov>. Help us make it better by
111 sharing your thoughts with us as you review the guide. If you adopt this solution for your own
112 organization, share your experience and advice with us. We recognize that technical solutions alone will
113 not fully enable the benefits of our solution, so we encourage organizations to share lessons learned and
114 best practices for transforming the business processes associated with implementing it.

115 ■ email dns-email-nccoe@nist.gov

116 ■ join our Community of Interest to offer your insights and expertise; email us at dns-email-nccoe@nist.gov

118 To learn more by arranging a demonstration of this reference solution, contacting us at dns-email-nccoe@nist.gov.

The National Cybersecurity Center of Excellence at the National Institute of Standards and Technology addresses businesses' most pressing cybersecurity problems with practical, standards-based example solutions using commercially available technologies. As the U.S. national lab for cybersecurity, the NCCoE seeks problems that are applicable to whole sectors, or across sectors. The center's work results in publicly available NIST Cybersecurity Practice Guides that provide modular, open, end-to-end reference designs.

LEARN MORE

<http://nccoe.nist.gov>

ARRANGE A DEMONSTRATION

nccoe@nist.gov

301-975-0200

NIST CYBERSECURITY PRACTICE GUIDE

DOMAIN NAME SYSTEMS-BASED ELECTRONIC MAIL SECURITY

Approach, Architecture, and Security Characteristics

For CIOs, CISOs, and Security Managers

Scott Rose

William Barker

Santos Jha

Chinedum Irrechukwu

Karen Waltermire

NIST SPECIAL PUBLICATION 1800-6B



DOMAIN NAME SYSTEMS-BASED ELECTRONIC MAIL SECURITY

1800-6B

Approach, Architecture, and Security Characteristics

For CIOs, CSOs, and Security Managers

Scott Rose

National Cybersecurity Center of Excellence
Information Technology Laboratory

William C. Barker
Dakota Consulting
Silver Spring, MD

Santos Jha
Chinedum Irrechukwu
The MITRE Corporation
McLean, VA

November 2016



U.S. Department of Commerce
Penny Pritzker, Secretary

National Institute of Standards and Technology
Willie May, Under Secretary of Commerce for Standards and Technology and Director

DISCLAIMER

Certain commercial entities, equipment, products, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by NIST or NCCoE, nor is it intended to imply that the entities, equipment, products, or materials are necessarily the best available for the purpose.

National Institute of Standards and Technology Special Publication 1800-6B
Natl Inst. Stand. Technol. Spec. Publ. 1800-6B, 73 pages (November 2016)
CODEN: NSPUE2

Organizations are encouraged to review all draft publications during public comment periods and provide feedback. All publications from NIST's National Cybersecurity Center of Excellence are available at <http://nccoe.nist.gov>.

Comments on this publication may be submitted to: dns-email-nccoe@nist.gov

Public comment period: November 2, 2016 through December 19, 2016

National Cybersecurity Center of Excellence
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, MD 20899
Mailstop 2002
Email: dns-email-nccoe@nist.gov

NATIONAL CYBERSECURITY CENTER OF EXCELLENCE

The National Cybersecurity Center of Excellence (NCCoE) at the National Institute of Standards and Technology (NIST) addresses businesses' most pressing cybersecurity problems with practical, standards-based solutions using commercially available technologies. The NCCoE collaborates with industry, academic, and government experts to build modular, open, end-to-end reference designs that are broadly applicable and repeatable. The center's work results in publicly available NIST Cybersecurity Practice Guides, Special Publication Series 1800, that provide users with the materials lists, configuration files, and other information they need to adopt a similar approach.

To learn more about the NCCoE, visit <http://nccoe.nist.gov>. To learn more about NIST, visit <http://www.nist.gov>.

NIST CYBERSECURITY PRACTICE GUIDES

NIST Cybersecurity Practice Guides (Special Publication Series 1800) target specific cybersecurity challenges in the public and private sectors. They are practical, user-friendly guides that facilitate the adoption of standards-based approaches to cybersecurity. They show members of the information security community how to implement example solutions that help them align more easily with relevant standards and best practices.

The documents in this series describe example implementations of cybersecurity practices that businesses and other organizations may voluntarily adopt. The documents in this series do not describe regulations or mandatory practices, nor do they carry statutory authority.

ABSTRACT

This document describes a security platform for trustworthy email exchanges across organizational boundaries. The project includes reliable authentication of mail servers, digital signature and encryption of email, and binding cryptographic key certificates to sources and servers. The example solutions and architectures presented here are based upon standards-based open-source and commercially available products.

KEYWORDS

authentication; data integrity; domain name system; digital signature; electronic mail; encryption; internet addresses; internet protocols; named entities; privacy

ACKNOWLEDGMENTS

We gratefully acknowledge the contributions of the following individuals and organizations for their generous contributions of expertise, time, and products.

Name	Organization
Nate Lesser	National Cybersecurity Center of Excellence
Karen Waltermire	National Cybersecurity Center of Excellence
Doug Montgomery	NIST ITL Advanced Networks Technology Division

Name	Organization
Janet Jones	Microsoft Corporation
Paul Fox	Microsoft Corporation
Joe Gersch	Secure64
Saksham Manchanda	Secure64
Benno Overeinder	NLnet Labs
Ralph Dolmans	NLnet Labs
Eillem Toorop	NLnet Labs
Bud Bruegger	Fraunhofer IAO
Victoria Risk	Internet Systems Consortium
Eddy Winstead	Internet Systems Consortium

Contents

1	Summary	1
1.1	The Challenge.....	3
1.2	The Solution.....	4
1.3	Benefits	5
1.4	Technology Partners and Collaborators	6
1.5	Feedback	6
2	How to Use This Guide	7
3	Introduction	9
4	Approach	11
4.1	Audience	12
4.2	DNS-Based Electronic Mail Security Project Scope	12
4.3	Assumptions	13
4.4	Risk Assessment	14
4.5	Technologies.....	32
5	Architecture	35
5.1	Usage Scenarios Supported	36
5.2	Architectural Overview	38
6	Outcome	46
6.1	The User's Experience.....	46
6.2	The System Administrator's Experience	51
7	Evaluation	53
7.1	Assumptions and Limitations	54
7.2	Testing	54
7.3	Scenarios and Findings	57
8	Future Build Considerations	60
Appendix A	Acronyms	61
Appendix B	References	63
Appendix C	DNS-Based Email Security Project Mapping to the Framework Core and Informative References	67

List of Figures

Figure 4.1	DNS-Based Email Security Collaborator Contributions	33
Figure 5.1	DNS-Based Email Security Deployment Diagram	38
Figure 5.2	DNS-Based Email Security Test Set-up	39
Figure 5.3	Fraudulent DNS Address Spoofing Configurations.....	41
Figure 5.4	Man-In-The-Middle Event Configurations	42

List of Tables

Table 5.1	Client Systems.....	43
Table 5.2	Mail Transfer Agents.....	44
Table 5.3	DNS Servers.....	44
Table 7.1	Tests Performed	55
Table C.1	PROTECT (PR).....	67
Table C.2	DETECT (DE).....	71
Table C.3	RESPOND (RS)	72

1 Summary

2	1.1	The Challenge	3
3	1.2	The Solution	4
4	1.3	Benefits	5
5	1.4	Technology Partners and Collaborators	6
6	1.5	Feedback	6
7			

This National Institute of Standards and Technology (NIST) Cybersecurity Practice Guide addresses the challenge of providing digital signature technologies to provide authentication and integrity protection for electronic mail (email) on an end-to-end basis, and confidentiality protection for email in transit between organizations. It implements and follows recommendations of NIST Special Publication 800-177 (SP 800-177), *Trustworthy Email*. Detailed protocol information and implementation details are provided in SP 800-177. Domain Name System¹ protection features are consistent with SP 800-81-2, *Secure Domain Name System (DNS) Deployment Guide*.

The NIST Special Publication 1800-6 series of documents contain:

- rationale for and descriptions of a Domain Name System-Based (DNS-Based) Electronic Mail (Email) Security platform that permits trustworthy email exchanges across organizational boundaries and
- a series of How-To Guides, including instructions for installation and configuration of the necessary services, that show system administrators and security engineers how to achieve similar outcomes

The solutions and architectures presented are built upon standards-based, commercially available products. These solutions can be used by any organization deploying email services that is willing to implement certificate-based cryptographic key management and DNS Security Extensions (DNSSEC)². Interoperable solutions are provided that are available from different types of sources (e.g., both commercial and open source products) and function in different operating systems environments.

This summary section describes the challenge addressed by this Volume B (*Approach, Architecture, and Security Characteristics*); describes the solution demonstrated to address the challenge; benefits of the demonstrated solution; lists the technology partners that participated in building, demonstrating, and documenting the solution; and explains how to provide feedback on this guide. [Section 2, How to Use This Guide](#) explains how each volume of the guide may be used by business decision makers, program managers, and Information Technology (IT) professionals such as systems administrators; and [Section 3, Introduction](#) provides a high-level project overview. [Section 4, Approach](#) provides a more detailed treatment of the scope of the project, describes the assumptions on which security platform development was based, describes the risk assessment that informed platform development, and describes the technologies and components that were provided by industry collaborators to enable platform development. [Section 5, Architecture](#) describes the usage scenarios supported by project security platforms, including Cybersecurity Framework³ functions supported by each collaborator-contributed component. [Section 6, Outcome](#) describes any changes in users' mail processing experience imposed by the additional security functionality, and summarizes changes to systems administrators' experiences with respect to integrating the new capabilities into their systems and in systems operations and maintenance. [Section 7, Evaluation](#) summarizes the test sequences that were employed to demonstrate security platform services, the Cybersecurity Framework functions to which each test sequence is relevant, the NIST SP 800-53-4 controls that applied to the functions being demonstrated, and an overview of

1. RFC 1591, *Domain Name System Structure and Delegation*
2. RFC 4033, *DNS Security Introduction and Requirements*
3. *Framework for Improving Critical Infrastructure Cybersecurity*, Version 1.0, National Institute of Standards and Technology February 12, 2014 <http://www.nist.gov/cyberframework/upload/cybersecurity-framework-021214.pdf>

49 platform performance in each of the two applications scenarios demonstrated. [Section 8](#),
50 [Future Build Considerations](#) is a brief treatment of other applications that might be explored in
51 the future in demonstrating the advantages of broader DNS security adoption. Appendices are
52 provided for acronyms, references, and a mapping of the DNS-Based Email Security project to
53 the Cybersecurity Framework Core⁴ and informative security references cited in the
54 Cybersecurity Framework Core.

55 1.1 The Challenge

56 Both private industry and the government are concerned about email security and the use of
57 email as an attack vector for cybercrime. Business operations are heavily reliant on email
58 exchanges and need to protect the confidentiality of business information, the integrity of
59 transactions, and privacy of individuals. Cryptographic services are used to authenticate the
60 source of email messages, protect against undetected unauthorized alteration of messages in
61 transit, and maintain message confidentiality. Efficiency and policies support reliance on mail
62 servers to provide cryptographic protection for email rather than on end-to-end security
63 operated by individual users. However, organizations need to protect their server-based email
64 security mechanisms against intrusion and man-in-the-middle attacks during automated
65 cryptographic service negotiation. In the absence of an appropriate combination of DNSSEC
66 and certificate-based protections, any of these attacks can result in disclosure or modification
67 of information by unauthorized third parties. The attacks can also enable an attacker to pose as
68 one of the parties to an email exchange and send email that contains links to malware-ridden
69 websites. If other content in a fraudulent message successfully motivates the user to click on
70 the link or the user's system is configured to automatically follow some links or download
71 content other than text, the malware will infect the user's system. Inclusion of links to malware
72 is a major factor in most confirmed data breaches. Consequences of such breaches can range
73 from exposure of sensitive or private information, to enabling fraudulent activity by the
74 attacker posing as the victimized user, to disabling or destroying the user's system-or that of the
75 user's parent organization. Beyond avoidance of negative consequences to users, improved
76 email security can also serve as a marketing discriminator for email service providers.

77 Implementation of DNSSEC and DNS-Based Authentication Of Named Entities (DANE)⁵ have
78 been impeded in the past by a shortage of easily used software libraries and by the fact that
79 most available email applications of the protocols respond to absent or incorrect digital
80 signatures by neither permitting delivery of the message nor alerting the mail server that
81 failure to deliver is based on a DNSSEC issue. The consequence of the first impediment is that,
82 unless forced by policy to do so, IT organizations defer DNSSEC/DANE implementation pending
83 availability of more mature software libraries. The consequence if the second is that, when
84 DNSSEC and DANE are turned on, mail servers experience severe service degradation or crashes
85 due to large numbers of retransmission attempts.

4. <http://www.nist.gov/cyberframework/>

5. RFC 6698, *The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security Protocol: TLSA*

1.2 The Solution

87 DNSSEC protects against unauthorized modifications to domain name information to prevent
 88 connection to spoofed or malicious hosts. The NCCoE initiated a collaborative project with
 89 industry partners to develop a proof-of-concept security platform that provides trustworthy
 90 mail server-to-mail server email exchanges across organizational boundaries. Products
 91 comprising the security platform include client mail user agents (MUAs)⁶, DNS servers
 92 (authoritative and caching/recursive)⁷, mail transfer agents, (MTAs)⁸, and X.509 cryptographic
 93 key certificate sources (components and services). The network infrastructure products are
 94 similar to those found in every enterprise and used to perform basic IT functions and handle
 95 email. The certificate utilities are needed to produce X.509 certificates⁹ for mail servers and
 96 end users to support Transport Layer Security (TLS)¹⁰ and Secure/Multipurpose Internet Mail
 97 Extensions (S/MIME)¹¹. This initial project focuses on Simple Mail Transfer Protocol (SMTP)¹²
 98 over TLS and S/MIME.

99 The DNS-based secure email building block project has demonstrated a security platform,
 100 consistent with SP 800-177, that provides trustworthy email exchanges across organizational
 101 boundaries. The project includes authentication of mail servers, digitally signing and encrypting
 102 email¹³, and binding cryptographic key certificates to the servers. The software library issue
 103 was addressed in SP 1800-6c by providing installation and configuration instructions for using
 104 and maintaining existing software libraries (including installation support applications). At the
 105 same time, inclusion of software developers and vendors in the development and
 106 demonstration process revealed software and implementation guidance shortcomings that
 107 have been corrected.

6. According to NIST Special Publication (SP) 800-177, an MUA is a software component (or web interface) that allows an end user to compose and send messages and to one or more recipients. An MUA transmits new messages to a server for further processing (either final delivery or transfer to another server).

7. According to Section 3.2 of SP 800-177, there are two main types of name servers: authoritative name servers and caching name servers. The term **authoritative** is with respect to a zone. If a name server is an authoritative source for DNS resource records for a particular zone (or zones) of DNS addresses, it is called an **authoritative name server** for that zone (or zones). An authoritative name server for a zone provides responses to name resolution queries for resources for that zone, using the records in its own zone file. A **caching name server** (also called a resolving/recursive name server), by contrast, provides responses either through a series of queries to authoritative name servers in the hierarchy of domains found in the name resolution query or from a cache of responses built by using previous queries.

8. Also according to SP 800-177, mail is transmitted, in a “store and forward” fashion, across networks via Mail Transfer Agents (MTAs). MTAs communicate using the Simple Mail Transfer Protocol (SMTP) described below and act as both client and server, depending on the situation.

9. RFC 5280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*

10. RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2*

11. RFC 5751, *Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification*

12. RFC 5321, *Simple Mail Transfer Protocol*

13. Cryptographic protection, while voluntary for the private sector has, for a number of applications been made mandatory for federal government agencies (see Managing Information as a Strategic Resource, OMB Circular A-130)

108 1.3 Benefits

109 Sectors across industries, as well as the federal government, are concerned about email
110 security and the use of email as an attack vector.¹⁴ Both public and private sector business
111 operations are heavily reliant on email exchanges. The need to protect the integrity of
112 transactions containing financial and other proprietary information and to protect the privacy
113 of employees and clients are among the factors that motivate organizations to secure their
114 email. Whether the service desired is authentication of the source of an email message,
115 assurance that the message has not been altered by an unauthorized party, or message
116 confidentiality, cryptographic functions are usually employed. Economies of scale and a need
117 for uniform implementation drive most enterprises to rely on mail servers to provide security to
118 the members of an enterprise rather than security implemented and operated by individual
119 users. Many server-based email security mechanisms are vulnerable to attacks involving:

120

- faked or fraudulent digital certificates
- otherwise invalid certificates
- failure to actually invoke a security process as a result of connection to or through a
123 fraudulent server

124 Even if there are protections in place, some attacks have been able to subvert email
125 communication by attacking the underlying support protocols such as DNS. Attackers can spoof
126 DNS responses to redirect email servers and alter email delivery. DNSSEC was developed to
127 prevent this. DNSSEC protects against unauthorized modifications to network management
128 information and host IP addresses. DNSSEC can also be used to provide an alternative
129 publication and trust infrastructure for service certificates using the DNS-based Authentication
130 of Named Entities (DANE) resource records.

131 The business value of the security platform that results from this project includes improved
132 privacy and security protections for users' communication, as well as improved management of
133 DNS and email security operations. Addressing the software library and message
134 retransmission issues, respectively, reduces the difficulty and cost of installing and maintaining
135 DNSSEC and DANE. Mitigating the major cause of system errors resulting from faulty
136 deployment of DNSSEC and DANE will encourage use of capabilities already present in many
137 email systems. Demonstration and publication of these improvements encourages wider
138 implementation of the protocols that provide Internet users with confidence that email has
139 been protected and reaches the intended receiver in a secure manner. The demonstrated
140 platform addresses three of the five core Functional Categories in the Framework for Improving
141 Critical Infrastructure Cybersecurity and many requirements of relevant security standards and
142 guidelines. Implementation of the platform will be increasingly important as a market
143 discriminator as public awareness of email security and privacy issues grows.

14. "How Cybercrime Exploits Digital Certificates," Infosec Institute, *General Security*, July 28, 2014, <http://resources.infosecinstitute.com/cybercrime-exploits-digital-certificates>

144 1.4 Technology Partners and Collaborators

145 The technology vendors who participated in this build submitted their capabilities in response
146 to a notice in the Federal Register. Companies with relevant products were invited to sign a
147 Cooperative Research and Development Agreement (CRADA) with NIST, allowing them to
148 participate in a consortium to build this example solution. We worked with:

- 149 ■ Microsoft Corporation
- 150 ■ NLnet Laboratories
- 151 ■ Secure64
- 152 ■ Internet Systems Consortium
- 153 ■ Fraunhofer IAO

154 1.5 Feedback

155 You can improve this guide by contributing feedback. As you review and adopt this solution for
156 your own organization, we ask you and your colleagues to share your experience and advice
157 with us.

- 158 ■ email dns-email-nccoe@nist.gov
- 159 ■ join our Community of Interest to offer your insights and expertise; email us at
160 dns-email-nccoe@nist.gov

161 Or learn more by arranging a demonstration of this example solution by contacting us at
162 dns-email-nccoe@nist.gov

2 How to Use This Guide

This NIST Cybersecurity Practice Guide demonstrates a standards-based reference design and provides users with the information they need to replicate this approach to email security. The reference design is modular and can be deployed in whole or in parts.

This guide contains three volumes:

- NIST SP 1800-6a: [Executive Summary](#)
- **NIST SP 1800-6b: Approach, Architecture, and Security Characteristics - what we built and why (you are here)**
- NIST SP 1800-6c: [How-To Guides](#) - instructions for building the example solution

Depending on your role in your organization, you might use this guide in different ways:

Business decision makers, including chief security and technology officers will be interested in the [Executive Summary](#) (NIST SP 1800-6a), which describes the:

- challenges enterprises face in implementing and operating a trustworthy email service
- example solution built at the NCCoE
- benefits of adopting the example solution

Technology or security program managers who are concerned with how to identify, understand, assess, and mitigate risk will be interested in this part of the guide. NIST SP 1800-6b describes what we did and why. [Section 4.4, Risk Assessment](#) will be of particular interest. This section provides a description of the risk analysis we performed and maps the security services provided by this example solution to the *Framework for Improving Critical Infrastructure Cybersecurity* and relevant security standards and guidelines.

You might share the [Executive Summary](#), NIST SP 1800-6a, with your leadership team members to help them understand the importance of adopting standards-based access management approaches to protect your organization's digital assets.

IT professionals who want to implement an approach like this will find the whole practice guide useful. You can use the [How-To Guides](#), NIST SP 1800-6c, to replicate all or parts of the build created in our lab. The How-To guide provides specific product installation, configuration, and integration instructions for implementing the example solution. We do not re-create the product manufacturers' documentation, which is generally widely available. Rather, we show how we incorporated the products together in our environment to create an example solution.

This guide assumes that IT professionals have experience implementing security products within enterprises. While we have used a suite of commercial and open source software products to address this challenge, this guide does not endorse these particular products. Your organization can adopt this solution or one that adheres to these guidelines in whole, or you can use this guide as a starting point for tailoring and implementing parts of a solution that would support the deployment of an trustworthy email system and the corresponding business

37 processes. Your organization's security experts should identify the products that will best
38 integrate with your existing tools and IT system infrastructure. We hope you will seek products
39 that are congruent with applicable standards and best practices. [Section 4.5, Technologies](#), lists
40 the products we used and maps them to the cybersecurity controls provided by this reference
41 solution.

42 A NIST Cybersecurity Practice Guide does not describe “the” solution, but a possible solution.
43 This is a draft guide. We seek feedback on its contents and welcome your input. Comments,
44 suggestions, and success stories will improve subsequent versions of this guide. Please
45 contribute your thoughts to dns-email-nccoe@nist.gov.

3 Introduction

As stated in section 1.1, both public and private sector business operations are heavily reliant on electronic mail (email) exchanges. They need to protect the integrity of transactions that may include financial and other proprietary information. The privacy of employees and clients is also a factor that motivates organizations to secure their email systems. Security services such as the authentication of the source of an email message, assurance that the message has not been altered by an unauthorized party, and confidentiality of message contents require the use of cryptographic functions. A need for uniform security implementation drives most enterprises to rely on mail servers to provide security to the members of an enterprise rather than rely on end users to implement a security policy on their own. However, most current server-based email security mechanisms are vulnerable to, and have been defeated by, attacks on the integrity of the cryptographic implementations on which they depend. The consequences frequently involve unauthorized parties being able to read or modify supposedly secure information, or to use email as a vector for inserting malware into the enterprise. Improved email security can help protect organizations and individuals against these consequences and also serve as a marketing discriminator for email service providers as well as improve the trustworthiness of enterprise email exchanges.

Domain Name System Security Extensions for the Domain Name System are technical mechanisms employed by domain owners to protect against unauthorized modification to network management information. DANE is a protocol that securely associates domain names with cryptographic certificates and related security information so that clients can better authenticate network services. In spite of the dangers of failure to authenticate the identities of network devices, adoption of DNSSEC has been slow. Demonstration of DANE-supported applications such as reliably secure email may support increased user demand for domain name system security. Follow-on projects might include HTTPS, IOT, IPSEC keys in DNS, and DNS service discovery.

The DNS-Based Email Security project demonstrated proof of concept security platforms composed of off the shelf components that provides trustworthy mail server-to-mail server email exchanges across organizational boundaries. The DANE protocol is used to authenticate servers and certificates in two roles in the DNS-Based Security for Email Project: (1) By binding the X.509 certificates used for Transport Layer Security (TLS) to DNSSEC signed names for mail server-to-mail server communication; and (2) by binding the X.509 certificates used for Secure/Multipurpose Internet Mail Extensions (S/MIME) to email addresses encoded as DNS names. These bindings support trust in the use of S/MIME certificates in the end-to-end email communication. The resulting platforms encrypt email traffic between servers and allow individual email users to obtain other users' certificates in order to validate signed email or send encrypted email.¹ The project will include an email sending policy consistent with a stated privacy policy that can be parsed by receiving servers so that receiving servers can apply the correct security checks.

1. S/MIME can do this now, but DANE makes it easier to actually use.

40 Documentation of the resulting platform includes statements of the security and privacy
41 policies and standards (e.g., Executive Orders, NIST standards and guidelines, IETF RFCs). This
42 also includes technical specifications for hardware and software, implementation
43 requirements, and a mapping of implementation requirements to the applicable policies,
44 standards, and best practices.

45 The secure email project has involved composition of a variety of components that were
46 provided by several different technology providers. Components include MUAs, DNSSEC
47 capable DNS servers, MTAs, and cryptographic certificate sources. These components are used
48 to generate and host DNSSEC signed zones and TLS enabled mail services.

49 This project resulted in demonstration of support to MUAs and MTAs by four DNS-based secure
50 email platforms and this publicly available NIST Cybersecurity Practice Guide that explains how
51 to employ the suite(s) to meet security and privacy requirements. This guide also provides
52 platform documentation necessary to compose a DNS-based email security platform from off
53 the shelf components that composed the prototype platforms.

4 Approach

2	4.1 Audience	12
3	4.2 DNS-Based Electronic Mail Security Project Scope	12
4	4.3 Assumptions.....	13
5	4.4 Risk Assessment.....	14
6	4.5 Technologies	32
7		

4.1 Audience

This guide is intended for individuals responsible for implementing security solutions in organizations' IT support activities. Current IT systems, particularly in the private sector often lack integrity protection for domain name services and electronic mail. The platforms demonstrated by the DNS-Based Email Security project, and the implementation information provided in these Practice Guides permit integration of DNS and email integrity services and email confidentiality services with minimum changes to existing infrastructure or impact to service operations. The technical components will appeal to system administrators, IT managers, IT security managers, and others directly involved in the secure and safe operation of the business IT networks.

4.2 DNS-Based Electronic Mail Security Project Scope

The DNS-Based Electronic Mail Security project is consistent with NIST SP 800-177 and demonstrates the use of off-the-shelf Transport Layer Security (TLS), Domain Name System (DNS) Security Extensions (DNSSEC), and DNS-based Authentication of Named Entities (DANE) components to achieve trustworthy electronic mail (email) objectives in a manner that is consistent with NIST SP 800-81-2.

4.2.1 Transport Layer Security (TLS)

The project uses TLS to protect confidentiality of email messages exchanged between mail servers. TLS relies on keys stored as X.509 digital certificates. These certificates can be used to authenticate the identity (server, domain or organization) of the certificate owner.

4.2.2 Domain Name System (DNS) Security Extensions (DNSSEC)

The project uses DNSSEC to authenticate and protect the integrity of DNS data. DNSSEC uses digital signatures over DNS data to prevent an attacker from tampering with or spoofing DNS responses. Mail servers use the DNS to find the destination of email as well as storing other artifacts necessary for email security (see below).

4.2.3 DNS-based Authentication of Named Entities (DANE)

The project uses DANE, a protocol that securely associates domain names with cryptographic certificates and related security information so that they can't be fraudulently modified or replaced to breach security. DNSSEC binds the X.509 certificates used for TLS to DNS.

4.2.4 Binding X.509 Certificates with DANE

The project also uses DANE to bind the X.509 certificates used for S/MIME to email addresses encoded as DNS names verified by DNSSEC.

⁴⁰ 4.2.5 Demonstration of Digital Signature and Encryption of Email

⁴¹ The project demonstrates sending encrypted messages between email systems resident in
⁴² different DNS domains, where the email exchanges between two organizations' email servers
⁴³ are carried over TLS, and the TLS key management is protected by DANE and DNSSEC. Signed
⁴⁴ email was sent between a message originator and a receiving party using end user applications
⁴⁵ (end-to-end) in different DNS domains, where the email exchanges between organizations
⁴⁶ were carried over TLS, the email messages were signed and verified with S/MIME on the
⁴⁷ end-users' client devices, and the S/MIME key management was protected by DANE and
⁴⁸ DNSSEC. In addition, the project demonstrated that the use of DNSSEC and DANE blocks an
⁴⁹ attempt by a fraudulent mail server to pose as the legitimate mail server for the receiver of the
⁵⁰ email.

⁵¹ 4.2.6 Demonstration of End-to-end Digital Signature of Mail

⁵² The project's digital signature demonstration included sending S/MIME signed email between a
⁵³ message originator and a receiving party using end user applications in different DNS domains.
⁵⁴ The email exchanges between organizations are carried over TLS, the email messages are
⁵⁵ signed and verified with S/MIME on the end-users' client devices, and the S/MIME certificates
⁵⁶ are stored in the DNS and protected by DNSSEC. This aspect of the project also demonstrated
⁵⁷ that use of DANE blocks an attempt by a fraudulent actor to pose as the email originators.

⁵⁸ 4.3 Assumptions

⁵⁹ 4.3.1 Security and Performance

⁶⁰ The email platforms and DNS services demonstrated provide email integrity and confidentiality
⁶¹ protection. An underlying assumption is that the benefits of using the demonstrated platforms
⁶² outweigh any additional performance risks that may be introduced. The security of existing
⁶³ systems and networks is out of scope for this project. A key assumption is that all potential
⁶⁴ adopters of one of the demonstrated builds, or any of their components, already have in place
⁶⁵ some degree of network security. Therefore, we focused on what potential new system
⁶⁶ vulnerabilities were being introduced to end users if they implement this solution. The goal of
⁶⁷ this solution is to not introduce additional vulnerabilities into existing systems, but there is
⁶⁸ always inherent risk when adding systems and adding new features into an existing system.

⁶⁹ 4.3.2 Modularity

⁷⁰ This assumption is based on one of the NCCoE core operating tenets. It is reasonably assumed
⁷¹ that organizations already have mail client and server systems in place. Our philosophy is that a
⁷² combination of certain components or a single component can improve email security for an
⁷³ organization; they may not need to remove or replace most existing infrastructure. This guide
⁷⁴ provides a complete top-to-bottom solution and is also intended to provide various options
⁷⁵ based on need.

76 4.3.3 Technical Implementation

77 This practice guide is written from a “how-to” perspective, and its foremost purpose is to
 78 provide details on how to install, configure, and integrate the components. The NCCoE assumes
 79 that an organization has the technical resources to implement all or parts of the build, or has
 80 access to companies that can perform the implementation on its behalf.

81 4.3.4 Operating System and Virtual Machine Environments

82 This project was conducted primarily in a Vmware vcenter server version 6.0.0 Build 3018523
 83 virtual machine environment. It is assumed that user organizations will be able to install the
 84 demonstrated applications in cloud-hosted VMs, local virtual machine or local native server
 85 client environments. This project uses Centos 7, Windows Server 2012R2, and Windows 10
 86 operating systems. Operating systems were chosen based on the requirements of the software.

87 The DNS-based secure email building block project assumes, and is dependent upon, the
 88 availability of off-the shelf information security technology. Particular products and expertise
 89 on which the project is dependent include those for MUAs, MTAs, DNS servers (authoritative
 90 and recursive) and X.509 certificate utilities.

91 4.4 Risk Assessment

92 According to NIST SP 800-30, *Risk Management Guide for Information Technology Systems*,
 93 “Risk is the net negative impact of the exercise of a vulnerability, considering both the
 94 probability and the impact of occurrence. Risk management is the process of identifying risk,
 95 assessing risk, and taking steps to reduce risk to an acceptable level.” The NCCoE recommends
 96 that any discussion of risk management, particularly at the enterprise level, begin with a
 97 comprehensive review of NIST 800-37, *Guide for Applying the Risk Management Framework to*
 98 *Federal Information Systems*. The risk management framework (RMF) and its associated
 99 references for identified security functions provides a baseline for organizing and relating to
 100 organizational objectives of:

- 101 1. the risks to electronic mail and the networks it transits
- 102 2. the security requirements to be met in order for the security platform to reduce these risks

103 While this guide does not present a full risk assessment, it does highlight the broad categories
 104 of threats and vulnerabilities associated with electronic mail.

105 4.4.1 Threats

106 Below are common threats associated with electronic mail:

- 107 ■ use of email as a vehicle for introducing malware
- 108 ■ use of email as a delivery mechanism for social engineering attacks
- 109 ■ theft or destruction of data communicated by email and/or its attachments due to loss or
 110 unauthorized/unintentional disposal of messages
- 111 ■ loss of privacy resulting from unauthorized access to email

112 ■ unauthorized modification of information communicated by email
 113 ■ malicious fraudulent creation of messages or attachments attributed to third parties

114 **4.4.2 Vulnerabilities**

115 Vulnerabilities are commonly associated with mail client applications, mail transfer
 116 applications, and network applications that are employed in creation, delivery, and reading of
 117 email. However, vulnerabilities can be exploited at all levels in the information stack. For
 118 up-to-date information regarding vulnerabilities, this guide recommends that security
 119 professionals leverage the National Vulnerability Database (NVD). The NVD is the U.S.
 120 government repository of standards-based vulnerability management data
 121 [<https://nvd.nist.gov>].

122 **4.4.2.1 Client System Vulnerabilities**

123 Organizations are getting better at protecting network perimeters, and companies with mature
 124 security programs usually allow only certain ports through the firewall and harden
 125 Internet-accessible servers to minimize the attack surface. As a result, attackers are paying
 126 closer attention to client-side vulnerabilities on internal workstations. These client-side
 127 vulnerabilities often are as simple as unpatched software on a desktop or laptop. Most client
 128 systems run at least one operating system and quite a few applications. Listing specific
 129 vulnerabilities for each is beyond the scope of this guide, but a current list of vulnerabilities and
 130 information regarding patches are available from NIST's National Vulnerability Database
 131 referenced above. Depending on the nature of a vulnerable application, an attacker may exploit
 132 it using a specially crafted email attachment or by convincing the user to visit a malicious Web
 133 site. Web browsers are common targets. Other attractive targets include Adobe Acrobat¹,
 134 Macromedia Flash², QuickTime³ and Java Runtime Environment⁴.

135 **4.4.2.2 Mail Server Vulnerabilities**

136 Mail servers have many of the same vulnerabilities as client systems, but we also need to be
 137 aware of protocol-based vulnerabilities involving access to valid lists of email addresses,
 138 vulnerabilities to relay exploits for malware insertion, vulnerabilities to email header
 139 disclosures, and vulnerabilities to viruses and worms. In the case of SMTP, one way that
 140 attackers can verify whether e-mail accounts exist on a server is simply to telnet to the server
 141 on port 25 and run the VRFY command.⁵ The VRFY command makes a server check whether a
 142 specific user ID exists. Spammers often automate this method to perform a **directory harvest**
 143 **attack**, which is a way of gleaning valid e-mail addresses from a server or domain for hackers to

1. See https://www.cvedetails.com/vulnerability-list/vendor_id-53/product_id-497/Adobe-Acrobat-Reader.html.

2. See
https://www.cvedetails.com/vulnerability-list/vendor_id-73/product_id-1950/version_id-8545/Macromedia-Flash-Player-6.0.29.0.html.

3. See <https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2015-7117>.

4. See <https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2015-4903>.

5. A number of ISPs now block port 25.

144 use. Scripting this attack can test thousands of e-mail address combinations. The SMTP
 145 command EXPN may allow attackers to verify what mailing lists exist on a server. Yet another
 146 way to capture valid e-mail addresses is to use applications such as *theHarvester* to glean
 147 addresses via Google and other search engines. In Microsoft Exchange, account enumeration is
 148 not generally an issue.

149 In environments other than Microsoft Exchange, account enumeration is not generally an issue.
 150 In such environments, the best solution for preventing this type of e-mail account enumeration
 151 depends on whether you need to enable commands like SMTP's VRFY and EXPN commands. In
 152 general, it is important to ensure that company e-mail addresses are not posted on the web.

153 Protocols like SMTP relay let users send e-mails through external servers. Open e-mail relays
 154 aren't the problem they used to be, but they can still be sources of vulnerabilities. Spammers
 155 and hackers can use an e-mail server to send spam or malware through e-mail under the guise
 156 of the unsuspecting open-relay owner.

157 In the case of email header disclosures, e-mail servers configured with typical defaults, may be
 158 vulnerable to divulging information such as internal IP addresses of e-mail clients, software
 159 versions of client and e-mail servers along with their vulnerabilities, or host names that can
 160 divulge network naming conventions

161 Email systems are regularly targeted by malware such as viruses and worms. It is necessary to
 162 verify that mail servers' antivirus software is actually working. As in the case of client systems
 163 vulnerabilities, NIST's National Vulnerability Database (<https://nvd.nist.gov>) is a frequently
 164 updated source of vulnerabilities that affect mail servers.

165 4.4.2.3 Network Vulnerabilities

166 The MITRE Corporation's Common Vulnerability Enumeration (CVE) lists more than 85,000
 167 vulnerabilities that can affect web servers, System Query Language (SQL) servers, DNS servers,
 168 firewalls, routers, and other network components (see <https://cve.mitre.org>). These include
 169 vulnerabilities to denial of service, code execution, overflow, cross-site scripting, directory
 170 traversal, process bypass, unauthorized gaining of information, SQL injection, file inclusion,
 171 memory corruption, cross-site request forgery, and http response splitting. Many of the
 172 vulnerabilities are operating system or applications-based. Others are protocol based (e.g.
 173 vulnerabilities inherent in IP⁶, TLS, DNS⁷, BGP⁸, SMTP and other network protocols). As in the
 174 case of client systems vulnerabilities, NIST's National Vulnerability Database
 175 (<https://nvd.nist.gov>) is a frequently updated source of vulnerabilities that affect network
 176 servers.

177 4.4.3 Risk

178 Risks are examined from the point of view of consequences of vulnerabilities being exploited.
 179 Some examples of these consequences include legal liability, consequences of failure to comply
 180 with regulations, confidentiality breaches, loss of productivity, and damage to organizational
 181 reputation.

6. RFC 791, *Internet Protocol*
7. RFC 1034, *Domain Names - Concepts And Facilities*
8. RFC 4271, *A Border Gateway Protocol 4 (BGP-4)*

182 ■ New and existing regulations are force organizations to keep a record of their emails and to
183 protect their employee and customer privacy. For example, the Health Insurance Portability
184 and Accountability Act (HIPAA) requires health care institutions to keep a record of their
185 email communications and secure confidentiality of information. In the new IRS regulation
186 Circular 230, the IRS requires tax advisors to add an email disclaimer to any emails including
187 tax advice, expressly stating that the opinion cannot be relied upon for penalty purposes.
188 The U.S. Securities and Exchange Commission and Gramm-Leach-Bliley Act impose similar
189 duties on financial institutions. Steep penalties can apply to those organizations that do not
190 comply with their industry's regulations. In a case lasting from 2000 until 2005, a
191 well-known financial institution was recently forced to pay 20 million dollars in penalties by
192 the Securities and Exchange Commission for not diligently searching for email back-up
193 tapes and over-writing multiple back-up tapes.

194 ■ Most confidentiality breaches occur from within the company. These breaches can be
195 accidental, but they can also be intentional.

196 ■ With respect to legal liability, organizations are generally held responsible for all the
197 information transmitted on or from their system, so inappropriate emails sent on the
198 company network can result in multi-million dollar penalties.

199 ■ Employees sending personal emails and sifting through spam mail can cause major loss of
200 productivity.⁹

201 ■ Even just a badly written email, or an email containing unprofessional remarks will cause
202 the recipient to gain a bad impression of the company that the sender is representing.
203 Fraudulent email attributable to an organization can do far more damage to an
204 organization's reputation, both in terms of the response elicited and in terms of loss of
205 confidence in the cybersecurity reliability of the organization

206 A number of cybersecurity actions are recommended to reduce these risks. The Framework
207 Core identified in NIST's *Framework for Improving Critical Infrastructure Cybersecurity* is a set of
208 cybersecurity activities, desired outcomes, and applicable references that are common across
209 critical infrastructure sectors. The Core presents industry standards, guidelines, and practices in
210 a manner that allows for communication of cybersecurity activities and outcomes across the
211 organization from the executive level to the implementation/operations level. The Framework
212 Core consists of five concurrent and continuous Functions: Identify, Protect, Detect, Respond,
213 and Recover. When considered together, these functions provide a high-level, strategic view of
214 the lifecycle of an organization's management of cybersecurity risk.

9. Current SPAM filtering solutions consist of some sort of filtering at the network or the PC level, and they don't reveal the details of the sender without looking up the source. It takes some work for the recipient. This will always put us one step behind the bad guys. DNS provides the necessary Internet-wide scaling and DNSSEC achieves this authentication.

4.4.4 Cybersecurity Framework Functions, Categories, and Subcategories Addressed by the DNS-Based Email Security Project

The *Framework for Improving Critical Infrastructure Cybersecurity*¹⁰ (Cybersecurity Framework) provides a common language for understanding, managing, and expressing cybersecurity risk both internally and externally. It can be used to help identify and prioritize actions for reducing cybersecurity risk, and it is a tool for aligning policy, business, and technological approaches to managing that risk. It can be used to manage cybersecurity risk across entire organizations or it can be focused on the delivery of critical services within an organization. Different types of entities - including sector coordinating structures, associations, and organizations - can use the Cybersecurity Framework for different purposes, including the creation of common Profiles. As stated above, the Framework Core provides a set of activities to achieve specific cybersecurity outcomes, and references examples of guidance to achieve those outcomes. The Core is not a checklist of actions to perform. It presents key cybersecurity outcomes identified by industry as helpful in managing cybersecurity risk. The Core comprises four elements: Functions, Categories, Subcategories, and Informative References.

- **Functions** organize basic cybersecurity activities at their highest level. These Functions are Identify, Protect, Detect, Respond, and Recover. They aid an organization in expressing its management of cybersecurity risk by organizing information, enabling risk management decisions, addressing threats, and improving by learning from previous activities. The Functions also align with existing methodologies for incident management and help show the impact of investments in cybersecurity. For example, investments in planning and exercises support timely response and recovery actions, resulting in reduced impact to the delivery of services.
- **Categories** are the subdivisions of a Function into groups of cybersecurity outcomes closely tied to programmatic needs and particular activities. Examples of Categories include "Asset Management," "Access Control," and "Detection Processes."
- **Subcategories** further divide a Category into specific outcomes of technical and/or management activities. They provide a set of results that, while not exhaustive, help support achievement of the outcomes in each Category. Examples of Subcategories include "External information systems are cataloged," "Data-at-rest is protected," and "Notifications from detection systems are investigated."
- **Informative References** are specific sections of standards, guidelines, and practices common among critical infrastructure sectors that illustrate a method to achieve the outcomes associated with each Subcategory. The Informative References presented in the Framework Core are illustrative and not exhaustive. They are based upon cross-sector guidance most frequently referenced during the Framework development process.

The DNS-Based E-Mail Security Building Block project supports the Cybersecurity Framework's Protect, Detect, and Respond Functions. Applicability to specific categories, subcategories, and functions is described in the following paragraphs.

10. *Framework for Improving Critical Infrastructure Cybersecurity*, Version 1.0, National Institute of Standards and Technology February 12, 2014. <http://www.nist.gov/cyberframework/upload/cybersecurity-framework-021214.pdf>

254 **4.4.4.1 Protect**

255 The Protect function develops and implements the appropriate safeguards needed to ensure
 256 delivery of critical infrastructure services. This function supports the ability to limit or contain
 257 the impact of a potential cybersecurity event. Examples of outcome Categories within this
 258 Function that are addressed by the DNS-Based E-Mail Security project include: Access Control
 259 and Protective Technology.

260 **1. Access Control (PR.AC)**

261 The Protect Function's Access Control Category supports an outcome in which access to
 262 assets and associated facilities is limited to authorized users, processes, or devices, and to
 263 authorized activities and transactions.

264 **a. PR.AC-1**

265 The **PR.AC-1** subcategory under Access Control supports identities and credentials
 266 being managed for authorized devices and users. The security platform resulting from
 267 the DNS-Based E-Mail Security project supports effective management of the
 268 credentials associated with the addresses from which electronic mail purportedly
 269 originates and the integrity of the user identities associated with the electronic mail.

270 The original design of the Domain Name System (DNS) did not include security; instead,
 271 it was designed to be a scalable distributed system. DNSSEC and DANE attempt to add
 272 security, while maintaining backward compatibility with the existing DNS. DNSSEC was
 273 designed to protect applications (and caching resolvers serving those applications) from
 274 using forged or manipulated DNS data. All answers from DNSSEC protected zones are
 275 cryptographically signed (i.e., digital signature over DNS data). By checking the digital
 276 signature, a DNS resolver is able to determine whether the information is authentic (i.e.
 277 unmodified and complete) and is served on an authoritative DNS server. While
 278 protecting IP addresses is the immediate concern for many users, DNSSEC can protect
 279 any data published in the DNS, including text records or mail exchange (MX) records,
 280 and can be used to bootstrap other security systems that publish references to
 281 cryptographic certificates stored in the DNS.

282 All DNSSEC responses contain signed DNS data. DNSSEC signature validation allows the
 283 use of potentially untrustworthy parties if (for example) the mail server is using a
 284 self-signed certificate. The protocol permit configuration of systems to accept messages
 285 whether or not they are digitally signed. The security platform developed under the
 286 DNS-Based E-Mail Security project permits electronic mail clients and transfer agents to
 287 be configured systems to send email messages to only server whose DNS entries are
 288 digitally signed. At the client systems level (e.g., Outlook, Postfix, Thunderbird), digital
 289 signature of the mail messages themselves can also be applied on user-to-user basis. In
 290 the user-to-user case, the signature provides assurance of the integrity of the identity of
 291 the sender rather than just the identity of the DNS zone(s) associated with the sender.

292 **b. PR.AC-3**

293 The **PR.AC-3** subcategory under Access Control supports management of remote
 294 access. One of the most common vectors for malware infection is a user clicking on a
 295 link that is included in an e-mail message from a spoofed source. Clicking on the link
 296 enables remote access to the user's system, and preventing delivery of e-mail from
 297 bogus sources represents a management control protecting against remote access by
 298 malicious entities. The DNS-Based E-Mail Security project's demonstrated security

299 platform can be used as a basis for accepting or refusing electronic mail based on
300 authenticated data stored in the DNS. This has an added benefit of supporting
301 protection against remote access based on other than e-mail functions.

302 **C. PR.AC-5**

303 The **PR.AC-5** subcategory under Access Control supports protection of network integrity
304 by incorporating network segregation where appropriate. The DNS-Based E-Mail
305 Security project does not employ specifically network segregation principles. However,
306 it does support network integrity by providing operationally feasible mechanisms for
307 preventing connections or message delivery to sources that do not implement a
308 specified set of DNS security extensions. Rigorous adherence to a minimum security
309 configuration can enforce effective isolation of a network from entities that do not
310 conform to the network's security requirements.

311 **2. Data Security (PR.DS)**

312 The Protect Function's Data Security Category supports an outcome in which information
313 and records (data) are managed consistent with the organization's risk strategy to protect
314 the confidentiality, integrity, and availability of information. The DNS-Based E-Mail Security
315 project demonstrates a capability to provide source and content integrity protection by
316 employing digital signature of messages and confidentiality protection by encrypting
317 messages.

318 **a. PR.DS-1**

319 The **PR.DS-1** subcategory under Data Security supports protection of data at rest. The
320 user-to-user digital signature capability demonstrated by the DNS-Based E-Mail Security
321 project can provide an ability to verify the source and content integrity of stored e-mail
322 messages where the digital signature is stored with the rest of the message. This
323 supports integrity protection for data-at-rest.

324 **b. PR.DS-2**

325 The **PR.DS-2** subcategory under Data Security supports protection of data in transit. In
326 addition to user-to-user digital signature of e-mail, the DNS-Based E-Mail Security
327 project demonstrates a capability to provide source and content integrity protection to
328 data-in-transit by employing server-to-server confidentiality protection to
329 data-in-transit by employing server-to-server encryption.

330 **C. PR.DS-6**

331 The **PR.DS-6** subcategory under Data Security supports use of integrity checking
332 mechanisms to verify software, firmware, and information integrity. The digital
333 signature of e-mail demonstrated by the DNS-Based E-Mail Security project's security
334 platform supports automatic integrity checking of information communicated in e-mail
335 messages. DNSSEC and DANE protect the integrity of address information.

336 **3. Protective Technology (PR.PT)**

337 The Protect Function's Protective Technology Category's goal is to ensure the security and
338 resilience of systems and assets by managing a technical security solution consistent with
339 related policies, procedures, and agreements.

340 a. **PR-PT-4**

341 The **PR.PT-4** subcategory under Protective Technology supports protection of
 342 communications and control networks. The DNS-Based E-Mail Security project
 343 demonstrates a capability to provide source and content integrity protection by
 344 employing digital signature of communications and confidentiality protection by
 345 encrypting communications. The support demonstrated for use of DNSSEC and DANE
 346 protocols also support communications and control network integrity by demonstrating
 347 operationally feasible mechanisms for refusing connections to or message delivery from
 348 sources that do not implement a specified set of DNS security extensions. Rigorous
 349 adherence to a minimum security configuration can be used to enforce isolation
 350 networks from entities that do not conform to the network's security requirements.

351 **4.4.4.2 Detect**

352 The Detect Function develops and implements the appropriate activities needed to identify in a
 353 timely manner the occurrence of a cybersecurity event. Examples of outcome categories within
 354 this function that are addressed by the DNS-Based E-Mail Security project include Security
 355 Continuous Monitoring and Detection Processes.

356 1. **Security Continuous Monitoring (DE.CM)**

357 The Security Continuous Monitoring Category supports an outcome in which information
 358 system and assets are monitored at discrete intervals to identify cybersecurity events and
 359 to verify the effectiveness of protective measures. While not a classic example of
 360 continuous monitoring, the DNS-Based E-Mail Security platform has the ability to
 361 automatically check all DNS responses for correct digital signatures.

362 a. **DE.CM-1**

363 The **DE.CM-1** subcategory under Security Continuous Monitoring supports monitoring
 364 of networks to detect potential cybersecurity events. While not a classic example of
 365 continuous monitoring, the demonstrated capability of the DNS-Based E-Mail Security
 366 platform to automatically check all inbound DNS responses for valid digital signatures
 367 permits identification of attempts to spoof systems using bogus DNS data. Automatic
 368 signing and signature validation for e-mail permits continuous checking for false sender
 369 identities and modification of message content.

370 b. **DE.CM-6**

371 The **DE.CM-6** subcategory under Security Continuous Monitoring supports monitoring
 372 of external service provider activity to detect potential cybersecurity events. While not
 373 a classic example of continuous monitoring, the demonstrated capability of the
 374 DNS-Based E-Mail Security platform to automatically check all inbound DNS responses
 375 for valid digital signatures permits detection and prevention of attempts by invalid service
 376 providers (e.g., bogus Certificate Authorities or Mail Transfer Agents) to spoof users'
 377 systems (including man-in-the-middle attacks).

378 2. **Detection Processes (DE.DP)**

379 The Detection Processes Category supports an outcome in which detection processes and
 380 procedures are maintained and tested to ensure timely and adequate awareness of
 381 anomalous events.

382 a. **DE.DP-4**

383 The **DE.DP-4** subcategory under Detection Processes supports event communication of
 384 detection information to appropriate parties. One of the shortcomings of most DNSSEC
 385 and DANE mechanisms is that they abort delivery of messages from sources whose
 386 DNSSEC signature checks fails to validate and do not provide any indication that failure
 387 is due to an invalid signature. This usually results in numerous retransmissions and
 388 consequent performance degradation or possible crashes. The DNS-Based E-Mail
 389 Security platform includes in its DNS resolvers notifications of DNS signature failures to
 390 mail agents in order to prevent consequent performance degradation. This
 391 communication of detection information has the potential to mitigate one of the
 392 primary impediments to private sector adoption of DNSSEC.

393 **4.4.4.3 Respond**

394 The Respond Function develops and implements the appropriate activities to take action
 395 regarding a detected cybersecurity event. This Function supports the ability to contain the
 396 impact of a potential cybersecurity event. Examples of outcome categories within this function
 397 that are addressed by the DNS-Based E-Mail Security project include: Response Planning,
 398 Communications, and Mitigation.

399 1. **Response Planning (RS.RP)**

400 The Response Planning Category supports an outcome in which response processes and
 401 procedures are executed and maintained, to ensure timely response to detected
 402 cybersecurity events.

403 a. **RS.RP-1**

404 The **RS.RP-1** subcategory under Response Planning supports execution of a response
 405 plan during or after an event. Inclusion of DNS and email security in security planning
 406 for systems connected to the Internet will necessarily include responses to detection of
 407 invalid digital signatures that include security flagging of connections and messages,
 408 and/or refusing connections and delivery of messages. Concurrent with detection of
 409 validation failure detection, these responses are demonstrated by the DNS-Based
 410 E-Mail Security platform.

411 2. **Communications (RS.CO)**

412 The RS.CO-2 subcategory under Communications supports reporting of events consistent
 413 with established criteria. As stated under DE.DP-4, one of the shortcomings of most DNSSEC
 414 and DANE mechanisms is that they abort delivery of messages to destinations whose
 415 DNSSEC signature checks fail but do not provide any indication that the failure is due to an
 416 invalid signature. In order to prevent consequent performance degradation, the DNS-Based
 417 E-Mail Security platform includes in its DNS resolver configuration notifications of DNSSEC
 418 signature failures to mail agents (i.e. configuration to log relevant DNSSEC issues). This
 419 communication of detection information has the potential to mitigate one of the primary
 420 impediments to private sector adoption of DNSSEC. It also provides a mechanism that can
 421 be exploited to provide to external stakeholders information involving failures of DNSSEC
 422 signature checks.

423 a. **RS.CO-2**

424 The **RS.CO-2** subcategory under Communications supports reporting of events
 425 consistent with established criteria. As stated under DE.DP-4, one of the shortcomings

426 of most DNSSEC and DANE mechanisms is that they abort delivery of messages to
 427 destinations whose DNSSEC signature checks fail but do not provide any indication that
 428 the failure is due to an invalid signature. In order to prevent consequent performance
 429 degradation, the DNS-Based E-Mail Security platform includes in its DNS resolvers
 430 notifications of DNSSEC signature failures to mail agents. This communication of
 431 detection information has the potential to mitigate one of the primary impediments to
 432 private sector adoption of DNSSEC. It also provides a mechanism that can be exploited
 433 to provide to external stakeholders information involving failures of DNSSEC signature
 434 checks.

435 **3. Mitigation (RS.MI)**

436 a. **RS.MI-1**

437 The **RS.MI-1** subcategory under Mitigation supports containment of incidents. In the
 438 case of incidents that compromise the integrity of network systems through which
 439 electronic mail is routed, the effects of the compromise can be limited to those local
 440 systems and devices that have not implemented the integrity and confidentiality
 441 mechanisms demonstrated by the DNS-Based E-Mail Security platform.¹¹

442 b. **RS.MI-2**

443 The **RS.MI-2** subcategory under Mitigation supports mitigation of incidents. The
 444 DNS-Based E-Mail Security project demonstrates user-to-user digital signature of
 445 messages. Retention of their digital signatures with stored messages permits later
 446 determination of whether the messages have been modified in storage. This can be a
 447 mitigating factor in the case of incidents that involve introduction of fraudulent
 448 information into electronic mail records. The project's demonstration of
 449 server-to-server encryption provides confidentiality protection for data-in-transit. This
 450 confidentiality protection can serve as a mitigating factor in the case of incidents
 451 involving unauthorized access to messages captured by network devices that sit
 452 between the sender's and recipient's mail servers.

453 **4.4.5 Cybersecurity References Directly Tied to those Cybersecurity**
 454 **Framework Categories and Subcategories Addressed by the**
 455 **DNS-Based Email Security Project**

456 The following security references were followed in accepting components for the DNS-Based
 457 Email Security platform, designing the platform, conducting demonstrations of the platform,
 458 and documenting the platform. The Framework functions, categories, and subcategories
 459 addressed by these references are listed for each reference. While many of the references were
 460 written as standards and guidelines to be applied to federal government agencies, their
 461 recommendations may also be applied in the private sector as best practices that support
 462 Cybersecurity Framework functional categories and subcategories. Those subcategories
 463 addressed by the DNS-Based Email Security platform are in **boldface**.

11. Note that, if a system is subverted, a lot of assumed security goes out the window. A subverted sending MTA could still be seen as valid by receivers for example.

464 1. *Security Requirements for Cryptographic Modules*, Federal Information Processing Standard
465 (FIPS), FIPS 140-2, May 2001. <http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf>

466 FIPS 140-2 provides a standard that is required to be used by Federal organizations when
467 these organizations specify that cryptographic-based security systems be used to provide
468 protection for sensitive or valuable data. Protection of a cryptographic module within a
469 security system is necessary to maintain the confidentiality and integrity of the information
470 protected by the module. All cryptographic components employed by the Federal
471 government outside the national security community, including NCCoE security platforms
472 that employ cryptography, must conform to FIPS 140-2. This standard specifies the security
473 requirements that will be satisfied by a cryptographic module. The standard provides four
474 increasing qualitative levels of security intended to cover a wide range of potential
475 applications and environments. The security requirements cover areas related to the secure
476 design and implementation of a cryptographic module. These areas include cryptographic
477 module specification; cryptographic module ports and interfaces; roles, services, and
478 authentication; finite state model; physical security; operational environment;
479 cryptographic key management; electromagnetic interference/electromagnetic
480 compatibility (EMI/EMC); self-tests; design assurance; and mitigation of other attacks.

481 Within the context of the Cybersecurity Framework, FIPS 140-2 provides standards for
482 “Protection” to be provided by cryptographic modules (PR.AC-2, **PR.AC-3**, PR.AC-4,
483 **PR.DS-1**, **PR.DS-2**, PR.DS-5, **PR.DS-6**, PR.IP-3, and **PR.PT-4**) and “Detection” of failures or
484 other exception conditions that might affect the protection afforded to systems by
485 cryptographic modules (**DE.CM-1**, DE.CM-2, and DM.DP-3).

486 2. *Guide for Applying the Risk Management Framework to Federal Information Systems: A
487 security Lifecycle Approach*, NIST Special Publication, SP 800-37 Rev. 1, Joint Task Force
488 Transformation Initiative; February 2010 with updates as of June 5, 2014.
489 <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-37r1.pdf>

490 SP 800-37 Rev. 1 provides guidelines for applying the Risk Management Framework (RMF)
491 to federal information systems. Systems to which the RMF is to be applied include NCCoE
492 use case and block activities. The RMF promotes the concept of near real-time risk
493 management and ongoing information system authorization through the implementation
494 of robust continuous monitoring processes; provides senior leaders with the necessary
495 information to make cost-effective, risk-based decisions with regard to the organizational
496 information systems supporting their core missions and business functions; and integrates
497 information security into the enterprise architecture and development life cycle. Applying
498 the RMF within enterprises links management processes at the information system level to
499 management processes at the organization level through a risk executive (function) and
500 establishes lines of responsibility and accountability for security controls deployed within
501 organizational information systems and inherited by those systems (i.e., common controls).

502 The six-step RMF includes security categorization, security control selection, security
503 control implementation, security control assessment, information system authorization,
504 and security control monitoring. With respect to the Cybersecurity Framework, SP 800-37
505 assumes that system components, business environment and governance structure have
506 been identified. The risk assessment that underlies categorization is based on the assumed
507 understanding of these factors. SP 800-37 also focuses on impacts of security incidents
508 rather than on threats that take advantage of system vulnerabilities to create those
509 impacts. The control selection, control implementation, and system authorization
510 recommendations of SP 800-37 do not map directly to the Cybersecurity Framework.

511 However, SP 800-37 does provide recommendations relevant to **Identify** (ID.RA-5, ID.RA-6,
 512 ID.RM 1, and ID.RM-2 in Section 3.1), **Protect** (PR.IP-3, and PR.IP-7 in Sections 3.4 and 3.6),
 513 and **Detect**, (DE.AE-5 and **DE.CM-1** in Section 3.6) elements of the Cybersecurity
 514 Framework.

515 3. *Guidelines on Electronic Mail Security*; NIST Special Publication; SP 800-45 Ver. 2; Tracy,
 516 Jansen, Scarfone, Butterfield; February 2007.

517 <http://csrc.nist.gov/publications/nistpubs/800-45-version2/SP800-45v2.pdf>

518 SP 800-45 provides guidelines intended to assist organizations in installing, configuring, and
 519 maintaining secure mail servers and mail clients. Specifically, the publication discusses in
 520 detail:

- 521 a. email standards and their security implications
- 522 b. email message signing and encryption standards
- 523 c. the planning and management of mail servers
- 524 d. securing the operating system underlying a mail server
- 525 e. mail-server application security
- 526 f. email-content filtering
- 527 g. email-specific considerations in the deployment and configuration of network
 528 protection mechanisms, such as firewalls, routers, switches, and intrusion detection
 529 and intrusion prevention systems
- 530 h. securing mail clients
- 531 i. administering the mail server in a secure manner, including backups, security

532 As suggested by its 2007 publication date, SP 800-45 doesn't reflect the most recent
 533 developments in electronic mail security, especially the more recent IETF RFCs (e.g.,
 534 SMIMEA¹² and TLSA¹³), but the recommendations it makes are still germane.

535 With respect to the Cybersecurity Framework's **Identify** category and its subcategories, SP
 536 800-45 recommends risk management activities, but does not go into detail that maps to
 537 subcategory references. In the **Protect** category, subcategory references **PR.AC-1**, **PR.AC-3**,
 538 **PR.AC-4**, **PR.AC-5**, **PR.AT-1**, **PR.AT-2**, **PR.AT-5**, **PR.DS-2**, **PR.DS-6**, **PR.IP-2**, **PR.IP-4**, and **PR.PT-1**
 539 are addressed by the guideline. In the **Detect** category, subcategory references **DP-1** and
 540 **DE.DP-4** are addressed by the guideline. In the **Respond** category, subcategory references
 541 **DE.AE-2**, **DE.CM-1**, **DE.CM-4**, **DE.CM-5**, **DE.CM-8**, **DE.DP-1**, and **DE.DP-4** are addressed. In
 542 the **Respond** category, subcategory references **RS.RP-1**, **RS.CO-1**, **RS.CO-2**, **RS.AN-1**, and
 543 **RS.IM-1** are addressed by the guideline. In the **Recover** category, subcategory reference
 544 **RC.RP-1** is addressed by the guideline.

545 4. *Federal S/MIME V3 Client Profile*, NIST Special Publication, SP 800-49, Chernick, November
 546 2002. <http://csrc.nist.gov/publications/nistpubs/800-49/sp800-49.pdf>

12. See *Using Secure DNS to Associate Certificates with Domain Names For S/MIME* ([draft-ietf-dane-smime-02](#)) and *Using Secure DNS to Associate Certificates with Domain Names For S/MIME* ([draft-ietf-dane-smime-12](#))

13. RFC 6698, The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA

SP 800-49 was developed to provide organizations with approaches to assure that Secure/Multipurpose Internet Mail Extensions (S/MIME) products can interoperate and meet the e-mail security needs of federal agencies both with respect to security features and adequate cryptographic algorithms. This profile states requirements for implementing sets of cryptographic algorithm suites specified elsewhere by the standards development organizations. The profile specifies a set of e-mail security features (e.g., encrypted e-mail and signed receipts) that are mandatory for federal agencies. SP 800-49 adds specificity to the S/MIME standards, while attempting to avoid violating those standards. As its 2002 publication date suggests, SP 800-49 is even more dated with respect to protocols than SP 800-45 (e.g., recommending the now deprecated SHA-1 instead of SHA-2 for hashing, and the deprecated Triple DES rather than AES for encryption). However, it too, makes security recommendations that are still germane. The SP 800-49 requirements and recommendations fall into the Cybersecurity Framework **Protect** category. It provides guidelines that address the subcategory references **PR.DS-2**, **PR.DS-6**, and (less precisely) **PR.PT-4**.

5. *Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations*; NIST Special Publication; SP 800-52 Rev. 1; Polk, McKay, Chokhani; April 2014. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-52r1.pdf>

Transport Layer Security (TLS) provides mechanisms to protect sensitive data during electronic dissemination across the Internet. SP 800-52 provides guidance in the selection and configuration of TLS protocol implementations, while making effective use of Federal Information Processing Standards (FIPS) and NIST-recommended cryptographic algorithms. SP 800-52 requires that TLS 1.1 be configured with FIPS-based cipher suites as the minimum appropriate secure transport protocol and recommended that agencies develop migration plans to TLS 1.2 by January 1, 2015. This Special Publication also identifies TLS extensions for which mandatory support must be provided and some other recommended extensions. Like SP 800-49, the SP 800-52 requirements and recommendations fall into the Cybersecurity Framework **Protect** category. The guideline addresses the subcategory references **PR.DS-2**, **PR.DS-6**, and (less precisely) **PR.PT-4**.

6. *Security and Privacy Controls For Federal Information Systems And Organizations*, NIST Special Publication, SP 800-53 Rev. 4, Joint Task Force Transformation Initiative, April 2013. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf>

SP 800-53 provides a catalog of security and privacy controls for federal information systems and organizations and a process for selecting controls to protect organizational operations (including mission, functions, image, and reputation), organizational assets, individuals, other organizations, and the nation from a diverse set of threats, including hostile cyberattacks, natural disasters, structural failures, and human errors. The controls are customizable and implemented as part of an organization-wide process that manages information security and privacy risk. The controls address a diverse set of security and privacy requirements across the federal government and critical infrastructure that are derived from legislation, Executive Orders, policies, directives, regulations, standards, and/or mission/business needs. The publication also describes how to develop specialized sets of controls, or overlays, that are tailored for specific types of missions/business functions, technologies, or environments of operation. Finally, the catalog of security controls addresses security from both a functionality perspective (the strength of security functions and mechanisms provided) and an assurance perspective (the measures of confidence in the implemented security capability). Addressing both security functionality

594 and security assurance ensures that information technology products and the information
 595 systems built from those products using sound systems and security engineering principles
 596 are sufficiently trustworthy.

597 SP 800-53 Rev. 4 addresses all Cybersecurity Framework categories and subcategories. Only
 598 the RC.CO-1 (Reputation after an event is repaired) and **RC.CO-2** (Recovery activities are
 599 communicated to internal stakeholders and executive and management teams) references
 600 under the **Recover**: Communications subcategory are not addressed by SP 800-53.

601 7. *Recommendation for Key Management: Part 1 - General*, NIST Special Publication 800-57
 602 Part Rev.4, Barker, January 2016; *Part 2 - Best Practices for Key Management Organization*,
 603 NIST Special Publication 800-57 Part 2, Barker, Barker, Burr, Polk, and Smid, August 2005;
 604 and *Part 3 - Application-Specific Key Management Guidance*, NIST Special Publication, SP
 605 800-57 Part 3 Rev. 1, Barker and Dang, January 2015.
 606 <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf>,
 607 <http://csrc.nist.gov/publications/nistpubs/800-57/SP800-57-Part2.pdf>,
 608 <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57Pt3r1.pdf>

609 NIST Special Publication 800-57 provides cryptographic key management guidance. Part 1
 610 provides general guidance and best practices for the management of cryptographic keying
 611 material. Part 2 provides guidance on policy and security planning requirements for U.S.
 612 government agencies. Part 3 of this Special Publication provides guidance when using the
 613 cryptographic features of current systems that may not exhibit all of the properties
 614 recommended by Part 1 of the guideline. Part 3 includes applications-specific
 615 recommendations for, among other applications, the Public Key Infrastructure (PKI),
 616 Internet Protocol Security (IPsec), Transport Layer Security (TLS) Secure/Multipart Internet
 617 Mail Extensions (S/MIME), and Domain Name System Security Extensions (DNSSEC). All of
 618 these recommendations apply directly to the DNS-Based E-Mail Security Building Block.

619 SP 800-57 addresses all of the Cybersecurity Framework categories except **Detect**. Audit is
 620 the primary mechanism relied on in SP 800-53 for detection purposes. The categories and
 621 subcategory references that are addressed by the guideline include Identify (ID.AM-2,
 622 ID.BE-3, ID.BE-4, ID.BE-5, ID.GV-1, ID.GV-4, ID.RA-4, and ID.RA-5), **Protect** (**PR.AC-1**, PR.AC-2,
 623 **PR.AC-3**, PR.AC-4, PR.AT-2, PR.AT-3, PR.AT-4, **PR.DS-1**, **PR.DS-2**, PR.DS-3, PR.DS-4, **PR.DS-6**,
 624 PR.IP-2, PR.IP-3, PR.IP-4, PR.IP-5, PR.IP-6, PR.IP-9, PR.PT-1, PR.PT-2, PR.PT-3, and **PR.PT-4**);
 625 **Respond** (**RS.RP-1**, RS.CO-1, **RS.CO-2**, RS.CO-3, RS.AN-2, and **RS.MI-2**); and **Recover**
 626 (RC.RP-1).

627 8. *Secure Domain Name System (DNS) Deployment Guide*, NIST Special Publication, SP
 628 800-81-2, Chandramouli and Rose, September 2013.
 629 <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-81-2.pdf>

630 The DNS is a distributed database that enables access to Internet resources via user-friendly
 631 domain names, rather than IP addresses, by translating domain names to IP addresses and
 632 back. The DNS infrastructure is made up of computing and communication entities called
 633 name servers, each of which contains information about a small portion of the domain
 634 name space. The name data provided by DNS is intended to be available to any computer
 635 located anywhere in the Internet. SP 800-81-2 provides deployment guidelines for securing
 636 DNS within an enterprise. The primary security goals for DNS are data integrity and source
 637 authentication, which are needed to ensure the authenticity of name information and
 638 maintain the integrity of name information in transit. This document provides extensive
 639 guidance on maintaining data integrity and performing source authentication. This

640 document presents guidelines for configuring DNS deployments to prevent many
 641 redirection attacks that exploit vulnerabilities in various DNS components.

642 The categories and subcategory references that are addressed are limited to **Identify**
 643 (ID.AM-2 and ID.RA-6), **Protect** (PR.AC-1, PR.AC-3, PR.AC-5, PR.AT-2, PR.DS-2, PR.DS-5,
 644 PR.DS-6, PR.IP-3, PR.IP-4, PR.IP-6, and PR.IP-9), and **Detect** (DE.CM-1 and DE.CM-7).

645 9. *A Framework for Designing Cryptographic Key Management Systems*; NIST Special
 646 Publication; SP 800-130; Barker, Branstad, Smid, Chokhani; August 2013.
 647 <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-130.pdf>

648 SP 800-130's framework for Designing Cryptographic Key Management Systems (CKMS)
 649 contains topics that should be considered by a CKMS designer when developing a CKMS
 650 design specification. For each topic, there are one or more documentation requirements
 651 that need to be addressed by the design specification. Thus, any CKMS that addresses each
 652 of these requirements would have a design specification that is compliant with this
 653 framework. A CKMS will be a part of a larger information system that executes processing
 654 applications. While the CKMS supports these applications by providing cryptographic key
 655 management services, the particular applications or particular classes of applications are
 656 beyond the scope of this framework.

657 SP 800-130 addresses all of the Cybersecurity Framework categories The categories and
 658 subcategory references that are addressed include **Identify** (ID.BE-4, ID.GV-1, ID.GV-2,
 659 ID.GV-3, ID.GV-4, ID.RA-1, ID.RA-2, ID.RA-3, ID.RA-5, and RM-1); **Protect** (PR.AC-1, PR.AC-2,
 660 PR.AC-4, PR.AC-5, PR.AT-1, PR.AT-2, PR.AT-4, PR.AT-5, PR.DS-1, PR.DS-2, PR.DS-3, PR.DS-6,
 661 PR.DS-7, PR.IP-1, PR.IP-3, PR.IP-4, PR.IP-5, PR.IP-6, PR.IP-9, PR.MA-1, PR.PT-1, PR.PT-2,
 662 PR.PT-3, and PR.PT-4); **Detect** (DE.AE-4, DE.CM-1, DE.CM-4, DE.CM-7, DE.CM-8, DE.DP-1,
 663 DE.DP-2, DE.DP-3, and DE.DP-5); **Respond** (RS.RP-1, RS.CO-1, RS.CO-2, RS.AN-2, RS.MI-1,
 664 and RS.MI-2); and **Recover** (RC.RP-1).

665 10. *A Profile for U.S. Federal Cryptographic Key Management Systems (CKMS)*; Third Draft; NIST
 666 Special Publication; SP 800-152; Barker, Smid, Branstad; December 18, 2014.
 667 <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-152.pdf>

668 Draft SP 800-152 covers major aspects of managing the cryptographic keys that protect
 669 federal information. Associated with each key is specific information (e.g., the owner
 670 identifier, its length, and acceptable uses) called metadata. The computers, software,
 671 modules, communications, and roles assumed by one or more authorized individuals when
 672 managing and using cryptographic key management services are collectively called a
 673 Cryptographic Key Management System (CKMS). The Profile for U. S. Federal Cryptographic
 674 Key Management Systems (FCKMSs) has been prepared to assist CKMS designers and
 675 implementers in selecting the features to be provided in their “products,” and to assist
 676 federal organizations and their contractors when procuring, installing, configuring,
 677 operating, and using FCKMSs.

678 SP 800-130 addresses all of the Cybersecurity Framework categories. The categories and
 679 subcategory references that are addressed include **Identify** (ID.AM-3, ID.AM-5, ID.BE-4,
 680 ID.BE-5, ID.GV-1, ID.GV-2, ID.GV-3, ID.GV-4, ID.RA-1, ID.RA-3, ID.RA-5, ID.RA-6, RM-1, and
 681 RM-2); **Protect** (PR.AC-1, PR.AC-2, PR.AC-3, PR.AC-4, PR.AC-5, PR.AT-1, PR.AT-2, PR.AT-4,
 682 PR.AT-5, PR.DS-1, PR.DS-2, PR.DS-3, PR.DS-4, PR.DS-6, PR.DS-7, PR.IP-1, PR.IP-3, PR.IP-4,
 683 PR.IP-5, PR.IP-6, PR.IP-7, PR.IP-8, PR.IP-9, PR.IP-12, PR.MA-1, PR.PT-1, PR.PT-2, PR.PT-3, and
 684 PR.PT-4); **Detect** (DE.AE-4, DE.CM-1, DE.CM-4, DE.CM-7, DE.CM-8, DE.DP-1, DE.DP-2,

685 DE.DP-3, and DE.DP-5); **Respond** (RS.RP-1, RS.CO-1, **RS.CO-2**, RS.AN-2, **RS.MI-1**, **RS.MI-2**,
686 RS.MI-3, and RS.IM-2); and **Recover** (RC.RP-1 and RC.IM-2).

687 11. *Trustworthy Email*; NIST Special Publication 800-177; Chandramouli, Garfinkle, Nightingale
688 and Rose; Draft Publication; September 2016.
689 <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-177.pdf>

690 NIST Special Publication 800-177 serves as a complimentary document to SP 800-45. SP
691 800-177 addresses email protocol security and provides descriptions, guidelines and
692 recommendations for deploying new email security protocols such as SMTP over TLS, email
693 supported by DANE, and other non-cryptographic authentication (e.g. Sender Policy
694 Framework, etc.). Discussions of SMTP over TLS and S/MIME relate directly to the work on
695 the DNS-Based Email Security Project builds.

696 With respect to the Cybersecurity Framework's Identify category and its subcategories, SP
697 800-177 recommends risk management activities, but does not go into detail that maps to
698 subcategory references. In the **Protect** category, subcategory references **PR.AC-1**, **PR.AC-3**,
699 **PR.AC-4**, **PR.AC-5**, **PR.AT-1**, **PR.AT-2**, **PR.AT-5**, **PR.DS-2**, **PR.DS-6**, **PR.IP-2**, **PR.IP-4**, and **PR.PT-1**
700 are addressed by the guideline. In the **Detect** category, subcategory references **DP-1** and
701 **DE.DP-4** are addressed by the guideline. In the **Respond** category, subcategory references
702 **DE.AE-2**, **DE.CM-1**, **DE.CM-4**, **DE.CM-5**, **DE.CM-8**, **DE.DP-1**, and **DE.DP-4** are addressed. In
703 the **Respond** category, subcategory references **RS.RP-1**, **RS.CO-1**, **RS.CO-2**, **RS.AN-1**, and
704 **RS.IM-1** are addressed by the guideline. In the **Recover** category, subcategory reference
705 **RC.RP-1** is addressed by the guideline.

706 4.4.6 Other Security References Applied in the Design and Development 707 of the DNS-Based Email Security Project

708 The following references provided additional security and protocol standards and guidelines
709 that were applied during design and development of the DNS-Based Email Security Project.

710 1. *Systems Security Engineering: An Integrated Approach to Building Trustworthy Resilient
711 Systems*, Draft, NIST Special Publication, SP 800-160, May 2014.
712 http://csrc.nist.gov/publications/drafts/800-160/sp800_160_second-draft.pdf

713 NIST Special Publication 160 defines system security engineering processes that are tightly
714 coupled to and fully integrated into well-established, international standards-based systems
715 and software engineering processes. The project supports the federal cybersecurity
716 strategy of “Build It Right, Continuously Monitor” and consists of a four-phase development
717 approach that will culminate in the publication of the final systems security engineering
718 guideline at the end of 2014. The four phases include:

719 a. **Phase 1:** Development of the system security engineering technical processes based on
720 the technical systems and software engineering processes defined in ISO/IEC/IEEE
721 15288:2008

722 b. **Phase 2:** Development of the remaining supporting appendices (i.e., Information
723 Security Risk Management (including the integration of the Risk Management
724 Framework [RMF], security controls, and other security- and risk-related concepts into
725 the systems security engineering processes), Use Case Scenarios, Roles and
726 Responsibilities, System Resiliency, Security and Trustworthiness, Acquisition

727 Considerations, and the Department of Defense Systems Engineering Process (Summer
 728 2014)

729 c. **Phase 3:** Development of the systems security engineering nontechnical processes
 730 based on the nontechnical systems and software engineering processes (i.e.,
 731 Agreement, Organizational Project-Enabling, and Project) defined in ISO/IEC/IEEE
 732 15288: 2008 (Fall 2014)

733 d. **Phase 4:** Alignment of the technical and nontechnical processes based on the updated
 734 systems and software engineering processes defined in ISO/IEC/IEEE DIS 15288:201x(E)
 735 (Fall or Winter 2014, subject to the final publication schedule of the international
 736 standards bodies)

737 The full integration of security engineering discipline into the systems and software
 738 engineering discipline involves fundamental changes in the traditional ways of doing
 739 business within organizations. This may involve breaking down institutional barriers that,
 740 over time, have isolated security activities from the mainstream organizational
 741 management and technical processes, including, for example, the system development life
 742 cycle, acquisition/procurement, and enterprise architecture. The integration of these
 743 interdisciplinary activities requires the strong support of senior leaders and executives, and
 744 increased levels of communication among all stakeholders who have an interest in, or are
 745 affected by, the systems being developed or enhanced.

746 2. *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*; IETF RFC 2459; Housley,
 747 Ford, Polk, Solo; January 1999. <https://www.rfc-editor.org/rfc/rfc2459.txt>

748 RFC 2459 is one part of a family of standards for the X.509 Public Key Infrastructure (PKI) for
 749 the Internet, but the RFC is a standalone document; implementations of this standard
 750 proceed independent from the other parts. The RFC profiles the format and semantics of
 751 public key certificates and certificate revocation lists for the Internet. Procedures are
 752 described for the processing of certification paths in the Internet environment. Encoding
 753 rules are provided for popular cryptographic algorithms. Finally, ASN.1 modules are
 754 provided in the appendices for all data structures defined or referenced.

755 3. *Threat Analysis of the Domain Name System (DNS)*, IETF RFC 3833, Atkins and Austein,
 756 August 2004. <https://tools.ietf.org/html/rfc3833>

757 RFC 3833 attempts to document some of the known threats to the DNS, and, in doing so,
 758 measure the extent to which DNSSEC is a useful tool in defending against these threats.

759 4. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL)
 760 Profile*; Proposed Standard; IETF RFC 5280; Cooper, Santesson, Farrell, Boeyen, Housley,
 761 Polk; May 2008. <https://datatracker.ietf.org/doc/rfc5280/>

762 RFC 5280 profiles the X.509 v3 certificate and X.509 v2 certificate revocation list (CRL) for
 763 use in the Internet. The RFC provides an overview and model of the specified approach,
 764 describes the X.509 v3 certificate format in detail, with additional information regarding the
 765 format and semantics of Internet name forms. Standard certificate extensions are
 766 described and two Internet-specific extensions are defined. A set of required certificate
 767 extensions is also specified, the X.509 v2 CRL format is described along with standard and
 768 Internet-specific extensions, an algorithm for X.509 certification path validation is
 769 described, and an ASN.1 module and examples are provided.

770 5. *Simple Mail Transfer Protocol*, IETF RFC 5321, Draft Standard, Kleinstein, October 2008.
 771 <https://tools.ietf.org/html/rfc5321>

772 RFC 5321 is a specification of the basic protocol for Internet electronic mail transport. It
 773 covers the SMTP extension mechanisms and best practices for the contemporary Internet,
 774 but does not provide details about particular extensions. Although SMTP was designed as a
 775 mail transport and delivery protocol, this specification also contains information that is
 776 important to its use as a “mail submission” protocol for “split-UA” (User Agent) mail reading
 777 systems and mobile environments.

778 6. *Secure/Multipurpose Internet Mail Extensions (S/MIME)*, Version 3.2, Message
 779 Specification, Proposed Standard, IETF RFC 5751, ISSN: 2070-1721, Ramsdell and Turner,
 780 January 2010. <https://tools.ietf.org/html/rfc5751>

781 RFC 5751 defines Secure/Multipurpose Internet Mail Extensions (S/MIME) version 3.2.
 782 S/MIME provides a consistent way to send and receive secure MIME data. The RFC
 783 describes methods for digital signatures to provide authentication, message integrity, and
 784 non-repudiation with proof of origin; encryption to provide data confidentiality; and to
 785 reduce data size.

786 7. *Use Cases and Requirements for DNS-Based Authentication of Named Entities (DANE)*, IETF
 787 RFC 6394, ISSN: 2070-1721, Barnes, October 2011. <https://tools.ietf.org/html/rfc6394>

788 Many current applications use the certificate-based authentication features in Transport
 789 Layer Security (TLS) to allow clients to verify that a connected server properly represents a
 790 desired domain name. Typically, this authentication has been based on PKI certificate chains
 791 rooted in well-known certificate authorities (CAs), but additional information can be
 792 provided via the DNS itself. This document describes a set of use cases in which the DNS and
 793 DNS Security Extensions (DNSSEC) could be used to make assertions that support the TLS
 794 authentication process. The main focus of this document is TLS server authentication, but it
 795 also covers TLS client authentication for applications where TLS clients are identified by
 796 domain names.

797 8. *The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security Protocol: TLSA*, Proposed Standard, IETF RFC 6698, ISSN: 2070-1721, Hoffman and Schlyter, August
 798 2012. <https://tools.ietf.org/html/rfc6698>

800 Encrypted communication on the Internet often uses Transport Layer Security (TLS), which
 801 depends on third parties to certify the keys used. RFC 6698 provides means to improve on
 802 that situation by standardizing on methods to enable the administrators of domain names
 803 to specify the keys used in that domain's TLS servers. This requires matching improvements
 804 in TLS client software, but no change in TLS server software.

805 9. *Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate
 806 Revocation List (CRL) Profile*, Proposed Standard, IETF RFC 6818, ISSN: 2070- 1721, Yee,
 807 January 2013. <https://tools.ietf.org/html/rfc6818>

808 RFC 6818 updates RFC 5280, the *Internet X.509 Public Key Infrastructure Certificate and
 809 Certificate Revocation List (CRL) Profile*. It changes the set of acceptable encoding methods
 810 for the explicit Text field of the user notice policy qualifier and clarifies the rules for
 811 converting internationalized name labels to ASCII. The RFC also provides some clarifications
 812 on the use of self-signed certificates, trust anchors, and some updated security
 813 considerations.

814 10. *SMTP security via opportunistic DANE TLS*, RFC 7672, Dukhovni and Hardaker, May 26, 2015.
 815 <https://tools.ietf.org/html/rfc7672>

816 The RFC describes a downgrade-resistant protocol for SMTP transport security between
 817 Message Transfer Agents (MTAs), based on the DNS-Based Authentication of Named
 818 Entities (DANE) TLSA DNS record. Adoption of this protocol will enable an incremental
 819 transition of the Internet email backbone to one using encrypted and authenticated
 820 Transport Layer Security (TLS).

821 11. *Using Secure DNS to Associate Certificates with Domain Names For S/MIME*, IETF Internet
 822 Draft Work in Progress, draft-ietf-dane-smime-12, Hoffman and Schlyter, July 31, 2016.
 823 <https://datatracker.ietf.org/doc/draft-ietf-dane-smime/>

824 The draft RFC for using secure DNS to associate certificates with domain names for S/MIME
 825 describes how to use secure DNS to associate an S/MIME user's certificate with the
 826 intended domain name; similar to the way that DANE (RFC 6698) does for TLS.

827 4.5 Technologies

828 The laboratory configuration employed for the DNS-Based Email Security project included
 829 components contributed by several sets of collaborating organizations. One of the component
 830 sets is Windows-based. The others are Linux-based. There were also three Mail User Agents
 831 (MUAs): Microsoft Outlook, Mozilla Thunderbird (on Linux), and a Thunderbird MUA equipped
 832 with a DANE-aware Apple Key Chain utility¹⁴ that were able to interact to all the mail servers via
 833 IMAP. While the Windows-based contribution used Server 2016 DNS services, the Linux-based
 834 contributions included three different implementations for DNS. One was based on NSD4 and
 835 Unbound authoritative and recursive servers, one was based on the Berkeley Internet Name
 836 Domain (BIND) DNS server, and one was based on the Secure64 DNS services. Secure 64 also
 837 contributed DNS services hosted on dedicated processors using SecureT micro O/S technology.
 838 Collaborators assisted in installation and initial configuration of products and, as necessary, in
 839 composition of components for different test cases.

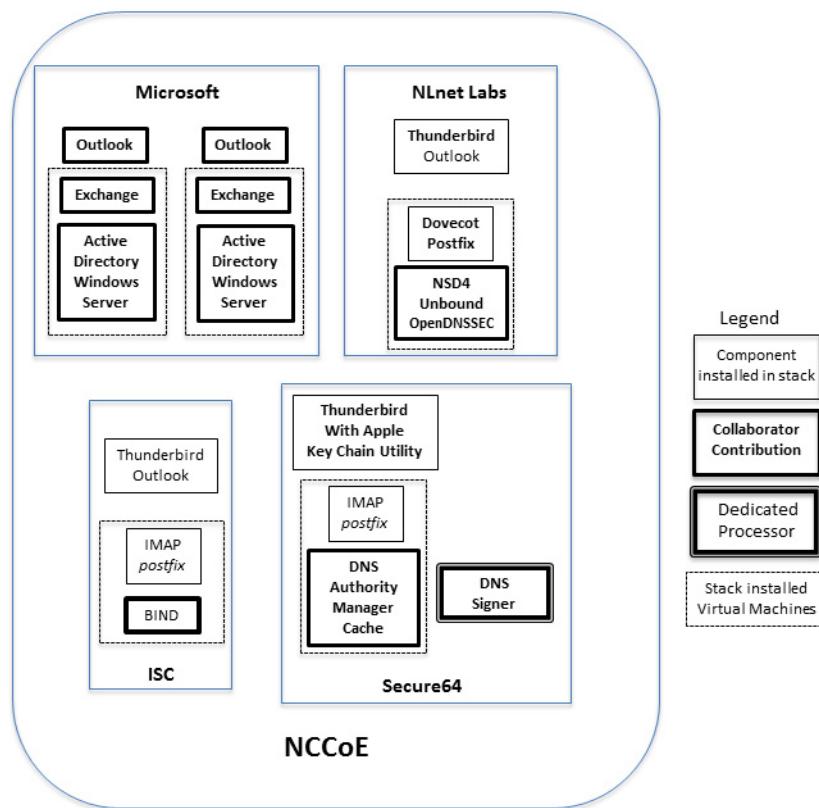
840 Figure 4.1 below depicts, at a high-level, collaborator contributions used to support the
 841 demonstration project. Elements identified in boldface are components provided or adapted by
 842 the collaborator. Other elements were incorporated into the stack to permit checking out the
 843 installed component's functionality.

844 Collaborator contributions identified below are organized with respect to the contributor as
 845 initially installed and checked out at the NCCoE. The architecture described in Section 5 below
 846 permits demonstration of the interconnection of components provided by different
 847 collaborators and initially checked out independently.

14. A utility for Public Key Retrieval into the Apple Key Chain. This utility is delivered on a Mac-
 Book loaded with Apple Mail and is a program for the MacBook that will fetch SMIMEA records
 and put them in the keystore so that we can demonstrate end-to-end security.

848

Figure 4.1 DNS-Based Email Security Collaborator Contributions



849

4.5.1 Microsoft

The Microsoft environments were contributed to support demonstration Scenario 1. Two environments were configured on the laboratory's VMware virtual machines (See figure 4.1 above). Each stack included the ability to demonstrate Office Outlook¹⁵ as an MUA, included Exchange Server 2016¹⁶ as MTAs, and used Active Directory running on Microsoft Windows Server 2016¹⁷ for DNS services. The Microsoft contribution included DNSSEC aware DNS recursive server, DNSSEC aware DNS authoritative server (IETF RFC 4033, 4034, and 4035), an MTA that can do SMTP over TLS (RFC 3207), management tools to configure servers and for debugging purposes, X.509 certificate sources, FIPS 140-2 validated cryptographic software, and support for multifactor authentication. The stacks were also able to be configured to demonstrate that Exchange could be used with either an Outlook or a Thunderbird MUA. Other test cases demonstrated using Exchange with a combination of other providers' DNS implementations.

15. https://en.wikipedia.org/wiki/Microsoft_Outlook

16. <https://products.office.com/en/exchange/microsoft-exchange-server-2016>

17. <https://www.microsoft.com/en-us/server-cloud/products/windows-server-2016/>

863 4.5.2 NLnet

864 The NLnet contribution focused on DNS services to support both demonstration scenarios.
 865 NLnet software was initially configured on the laboratory's VMware virtual machines. The
 866 components included NSD4 4.1.9¹⁸, Unbound¹⁹, and OpenDNSSEC²⁰ software for DNS services
 867 and Postfix and Dovecot for mail services. NSD4 is an authoritative only, high performance,
 868 open source name server. Unbound is a validating, recursive, caching DNS resolver.
 869 OpenDNSSEC is a set of software for signing DNS zones that are then served using NSD. While
 870 OpenDNSSEC can be configured to sign zone files or to sign zones transferred in via DNS zone
 871 transfer (AXFR), in these scenarios, it is used to sign local zone files in these scenarios. Like with
 872 the Microsoft stack above, multiple MUAs were configured to send and receive mail with the
 873 NLnet components via SMTP and IMAP.

874 4.5.3 ISC

875 The ISC contribution was focused on the BIND DNS server and supported both demonstration
 876 scenarios. BIND was initially configured on the laboratory's VMware virtual machines and
 877 included configuration for Postfix and Dovecot for email. BIND²¹ is open source software that is
 878 considered the reference implementation of DNS, but it is also production-grade software,
 879 suitable for use in high-volume and high-reliability applications. BIND features response rate
 880 limiting (RRL), support for FIPS 140-2 validated hardware cryptographic modules, the optional
 881 ability to retrieve zone data directly from an external database, the ability to use in-line signing
 882 to automatically re-sign records as they are updated, and a scalable master/slave hierarchy. Like
 883 the other stacks, all three MUAs were able to connect and use the stack for DNS and email.

884 4.5.4 Secure64

885 The Secure64 contributions were focused on DNS services to support both demonstration
 886 scenarios. The Secure64 environment included an automated online Secure64 DNS Signer as
 887 well as DNSSEC-capable VM images of DNS Cache, DNS Authority, and DNS Manager. DNS
 888 Manager provided centralized management of Secure64 DNS Cache software and
 889 configurations and provided network-wide monitoring of key performance indicators. DNS
 890 Manager allowed creation of groups of servers and assignment of configurations to a group, a
 891 single server, or all servers. DNS Authority is an authoritative signer and server as a single
 892 platform. DNS Cache, DNS Authority, and DNS Manager were configured on the laboratory's
 893 VMware virtual machine; and the DNS Signer was provided as a high-assurance implementation
 894 delivered on a Secure64 dedicated appliance. Secure64 contributions were able to demonstrate
 895 Outlook, Thunderbird, or Thunderbird equipped with an Apple Key Chain utility as MUAs and
 896 use Postfix as an MTA and Dovecot to provide IMAP for clients.

18. <http://www.nlnetlabs.nl/projects/nsd/>

19. <http://unbound.net>

20. <https://www.opendnssec.org>

21. <https://www.isc.org/downloads/bind/>

5 Architecture

2	5.1 Usage Scenarios Supported	36
3	5.2 Architectural Overview	38
4		

5 The Security platform architecture used for the DNS-Based Email Security project included
 6 combinations of components from different sources that supported two usage scenarios for
 7 DANE-enabled secure email in four different systems environments.

8 .5.1 Usage Scenarios Supported

9 The scenarios supported include:

- 10 ■ “ordinary” email where the email exchanges between two organizations’ email servers
 11 communicate over TLS with a STARTTLS extension, and relevant TSLA records are published
 12 in the receiver’s DNS zone protected by DNSSEC; and
- 13 ■ end-to-end signed email, where the email exchanges between users in different
 14 organizations are carried over a channel protected by TLS (using the STARTTLS extension),
 15 and relevant artifacts used for signing and channel protection are published in a DNS zone
 16 protected by DNSSEC. Subsequently, these artifacts are used for S/MIME and TLS validation.

17 In both scenarios, end-entity and personal certificates were generated from Certificate
 18 Authorities (CAs). Use of “well known” (i.e. installed as trust anchors in hosts), local enterprise
 19 CAs, and self-signed certificates were demonstrated.

20 While the second scenario demonstrated signing of emails, it does not include an end-to-end
 21 encrypted email scenario. Signing addresses the main security concerns in enterprise
 22 environments, which are the target of the project, but may neglect concerns of individual users
 23 who may also want to reduce information disclosure to their email providers. The two
 24 scenarios that are included may, however, serve as enablers for end-to-end encryption.
 25 Participation by parties having a primarily end-to-end encryption focus may succeed in
 26 generating industry support for the building blocks needed to support end-to-end encryption.

27 In more detail, the project’s security platforms use the STARTTLS extension to include
 28 encryption of communications between two MTAs, as well as the signature of individual
 29 messages using S/MIME. The encryption and decryption with S/MIME on the end user’s client
 30 was excluded from the current platform demonstration.

31 5.1.1 Usage Scenario 1

32 An individual needs to enter into an email exchange with an individual in another organization
 33 Each individual exchanges email via the respective parent organizations’ mail servers. Users
 34 connect to their organizations’ respective mail servers within a physically protected zone of
 35 control.

36 In this scenario, the privacy policy of the parent organizations requires encryption of the
 37 information being exchanged. The security afforded by the cryptographic process is dependent
 38 on the confidentiality of encryption keys from unauthorized parties. The mail servers are
 39 configured to use X.509 certificates to authenticate themselves during an encryption key
 40 establishment process.

41 DNSSEC is employed to ensure that each sending mail server connects to the legitimate and
42 authorized receiving mail server from which its X.509 certificate is obtained. DANE resource
43 records are employed to bind the cryptographic keying material to the appropriate server
44 name. STARTTLS is employed to negotiate the cryptographic algorithm to be employed with TLS
45 in the email exchange in which the PII is transferred. Encryption of the email message is
46 accomplished by the originator's email server, and decryption of the email message is
47 accomplished by the recipient's email server.

48 Demonstrations of the security platform in this scenario include an attempt by a fraudulent
49 mail server to pose as the legitimate receiver of the email and a man-in-the-middle attacker to
50 attempt to disrupt the signal that TLS is available for the desired destination. In the latter
51 attack, the goal is to force unencrypted transmission of the email. Both attempts should fail due
52 to use of DNSSEC and DANE.

53 5.1.2 Usage Scenario 2

54 An individual needs to enter into an email exchange with an individual in another organization.
55 Each individual exchanges email via the respective parent organizations' mail servers. Users
56 connect to their organizations' respective mail servers within a physically protected zone of
57 control.

58 The policy of the parent organizations requires cryptographic digital signature of the message
59 to provide integrity protection source authentication of the email message. S/MIME is a widely
60 available and used protocol for digitally signing electronic mail. Each organization has therefore
61 generated X.509 certificates for their users that include the public portion of their signature
62 keys. These certificates are then published in the DNS using the appropriate DANE DNS
63 Resource Record (RR) type.

64 DNSSEC is used to provide assurance that the originating user's mail server connects to the
65 intended recipient's mail server. DANE records are employed to bind the cryptographic
66 certificates to the appropriate server (for TLS) and individual user (for S/MIME), respectively.
67 TLS is employed to provide confidentiality. Digital signature of the email message is
68 accomplished by the originator's email client. Validating the signature (hence the integrity of
69 the authorization provided in the email message) is accomplished by the recipient's email
70 client.

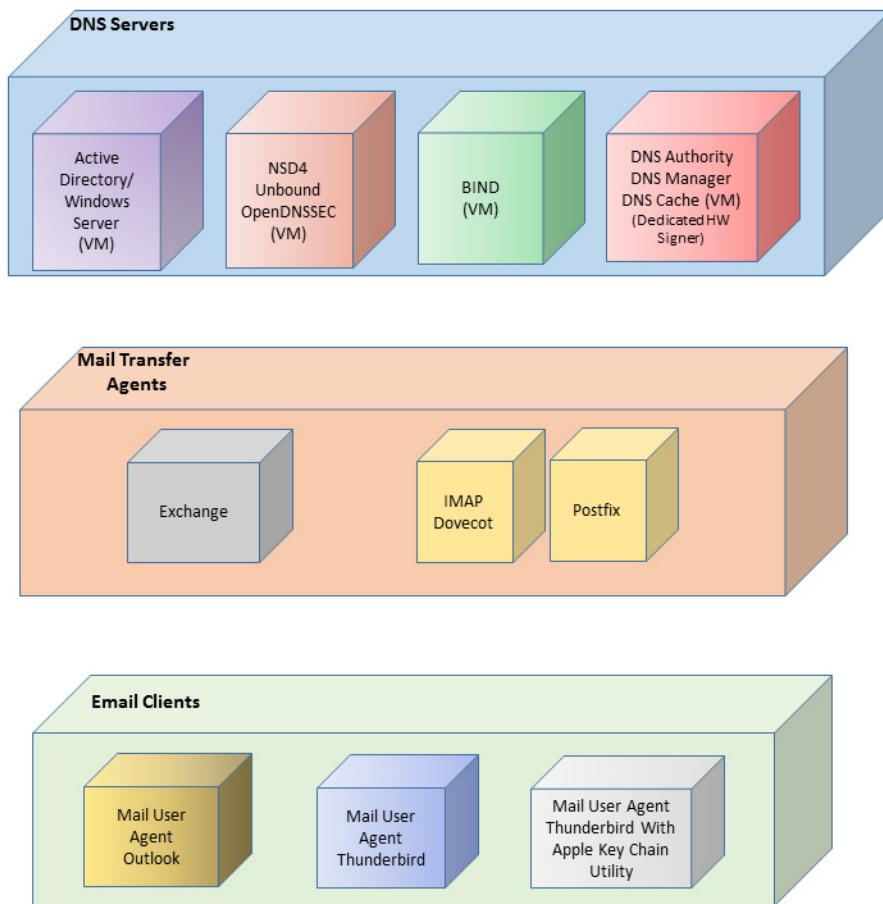
71 Demonstrations of the security platform in this scenario include an attempt by a fraudulent
72 actor to pose as the originator of the email and a man-in-the-middle attacker attempting to
73 disrupt the validation the S/MIME signature. Both attempts fail due to use of DNSSEC and DANE
74 records.

5.2 Architectural Overview

The laboratory architecture for the DNSSEC-Based Email Security project was designed to permit interconnection of Microsoft Outlook and Thunderbird MUAs with Microsoft Exchange and Postfix/Dovecot MTAs. It demonstrates the interconnection of either MTA with any of the DNS services contributed by collaborators. Two instantiations of each MTA type were established to demonstrate email exchanges between MTAs of the same type or different types. The various component combinations are then demonstrated with three different TLSA RR parameters: a self-signed certificate, use of local certificate authorities, and use of well-known certificate authorities.

Figure 5.1 is a deployment diagram of the architecture used for demonstrating DNS-Based Email Security.

Figure 5.1 DNS-Based Email Security Deployment Diagram



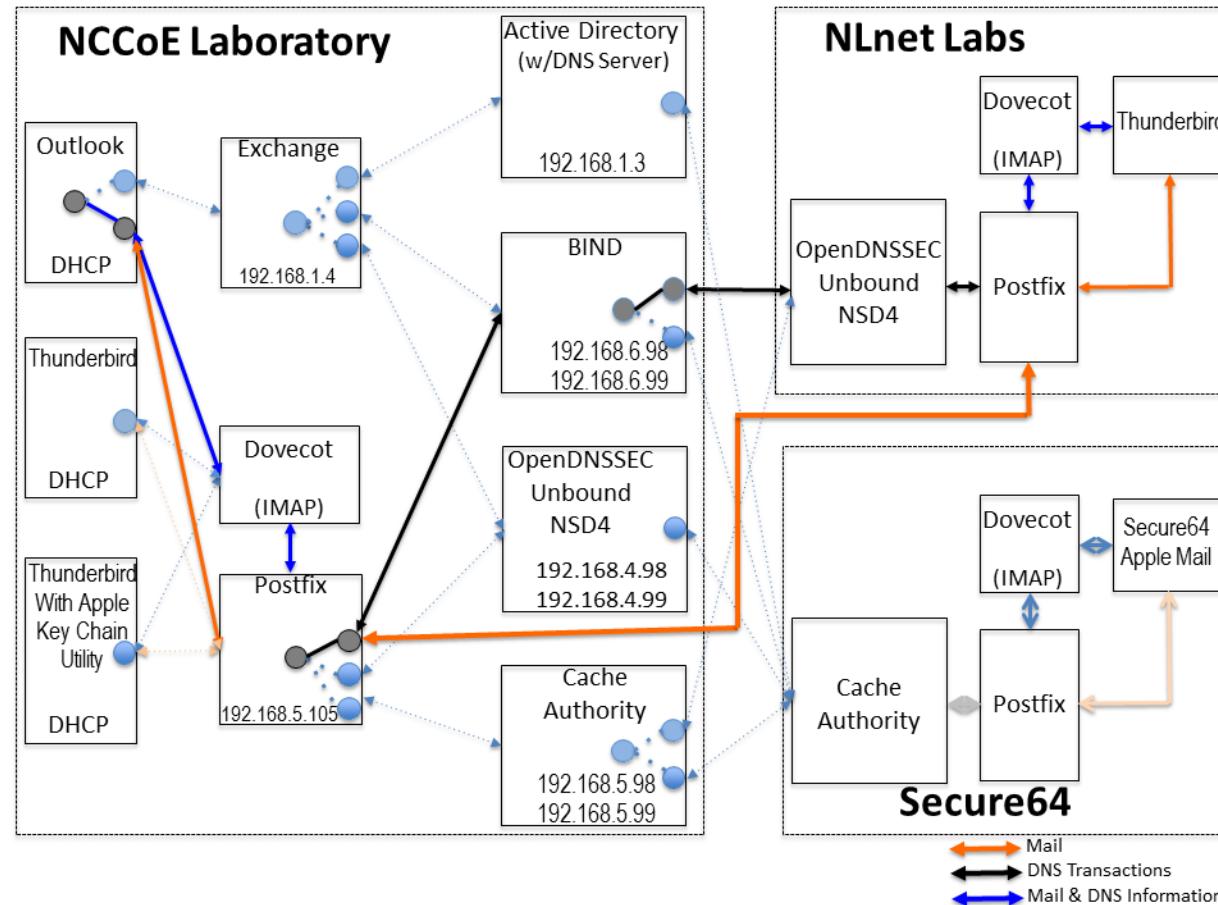
For test documentation purposes, the receiving MTA is named differently depending on the receiver's DNS service zone and the TLSA option being demonstrated. The sending MTA's implementation and DNS infrastructure can also vary for each test, but share the same basic processes.

92

The design of the environment permits interconnection of components provided by different collaborators (see [figure 5.2](#)).

93

Figure 5.2 DNS-Based Email Security Test Set-up



94

The depiction shows that the project security platform test/demonstration activity was based on three different clients, two MTAs, and four DNS service configurations in the lab at the NCCoE exchanging messages with NLnet Labs and Secure64. All messages were signed (a mail client function) and encrypted (server to server). We worked with one remote location at a time, driven by whichever is ready first. The message exchanges, including DNS activity will be logged at each end (lab and remote correspondent).

99 The solid connectors in the depiction illustrate one case. The dotted lines depict the other cases
100 we'll want to demonstrate. A switch convention is used to reflect configuration options, but the
101 project team actually configures each component for each option.

102 The orange arrows between the mail clients and the Postfix MTA reflect the fact that clients
103 submitted email directly to the SMTP server for relay, while using Dovecot only to get mail. (The
104 depiction in [figure 5.2](#) reflects that IMAP isn't used to submit mail, only retrieve it, so the MUA
105 sent mail directly to the Postfix server, but received the reply through the Dovecot server.)

106 The project team demonstrated 30 different events using various combinations of MUA, MTA,
107 and DNS Server components divided among five test sequences. In each sequence, signed and
108 encrypted messages were sent from a sender to a recipient. Both Exchange and Postfix
109 encrypted mail by default. Most of the exchanges employed either self-signed certificates or
110 local CAs (see [Appendix C](#)). The BIND configuration was set up to obtain and validate
111 certificates from the NIST Advanced Networks Technology Division's (ANTD's) DNS source
112 (acting as a root CA). (See [section 7](#) below for test sequence sets.) Both Exchange and Postfix
113 encrypted mail by default. Most of the exchanges employed either self-signed certificates or
114 local CAs.

115 In one test sequence, fraudulently S/MIME signed email was sent from a malicious sender to
116 recipients using Outlook and Thunderbird MUAs configured to use Exchange and Postfix as
117 MTAs. The Outlook/Exchange configuration used Active Directory as its DNS server. The
118 configurations employing Postfix/Dovecot MTAs were demonstrated with each of the other
119 three contributed DNS Services. In one event, the Thunderbird MUA employed an Apple Key
120 Chain Utility tool that allows a host to obtain X.509 certificates via of DANE RRs. All events were
121 conducted using well-known CA and Enterprise CA-issued certificates for the impersonated
122 sender. The fraudulent site attempted to spoof a valid sending domain belonging to a Secure64
123 site that was configured with DNS Authority/Cache/Signer DNS services, a Postfix/Dovecot
124 MTA, and Thunderbird equipped with the Apple Key Chain utility. An Outlook/Exchange/ Active
125 Directory set-up acted as the fraudulent site. The email exchange between organizations was
126 carried over TLS, and the email message was S/MIME signed on the fraudulent users' client
127 device. The set-up for this sequence is depicted in [figure 5.3](#) below.

128 In another sequence, an NCCoE system attempted to send a TLS protected email from Exchange
129 and Postfix MTAs (in turn) to an external Postfix MTA using DNS Authority/Cache/Signer for
130 DNS services. The NCCoE Exchange MTA used Active Directory DNS Services, and the
131 Postfix/Dovecot MTA used BIND and NSD4/Unbound/OpenDNSSEC DNS services. A S/MIME
132 signed email was sent to an external Postfix MTA. Four events were conducted using
133 Well-Known CA issued certificates, four events were conducted using Enterprise CA issued
134 certificates (TLSA/SMIMEA RR parameter of CU=2) for TLS and S/MIME on the receiver side,
135 and three events were conducted using self-signed certificates (TLSA/SMIMEA RR parameter of
136 CU=3) for TLS and S/MIME on the receiver side. An Outlook/Exchange/Active Directory stack
137 acted as a man-in-the-middle and attempted to intercept the message. [Figure 5.4](#) depicts the
138 configuration for a man-in-the-middle demonstration. Note that the sender is being
139 misdirected to a malicious email server only. This is to simulate a lower level attack where email
140 is sent (via route hijacking or similar low level attack) to a Man-in-the-Middle. [Figure 5.4](#)
141 depicts the configurations used with the Thunderbird/Postfix/ Dovecot/Bind option selected.

Figure 5.3 Fraudulent DNS Address Spoofing Configurations

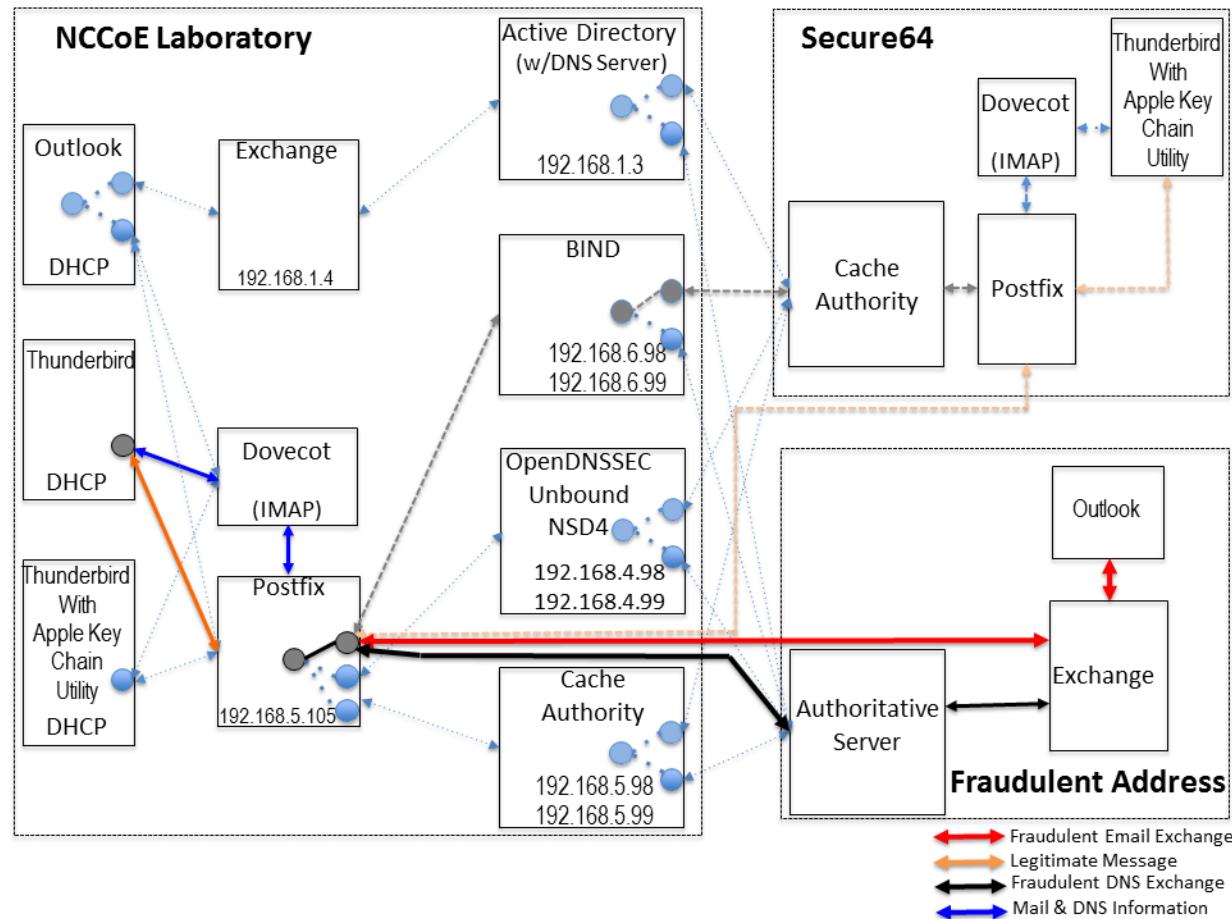
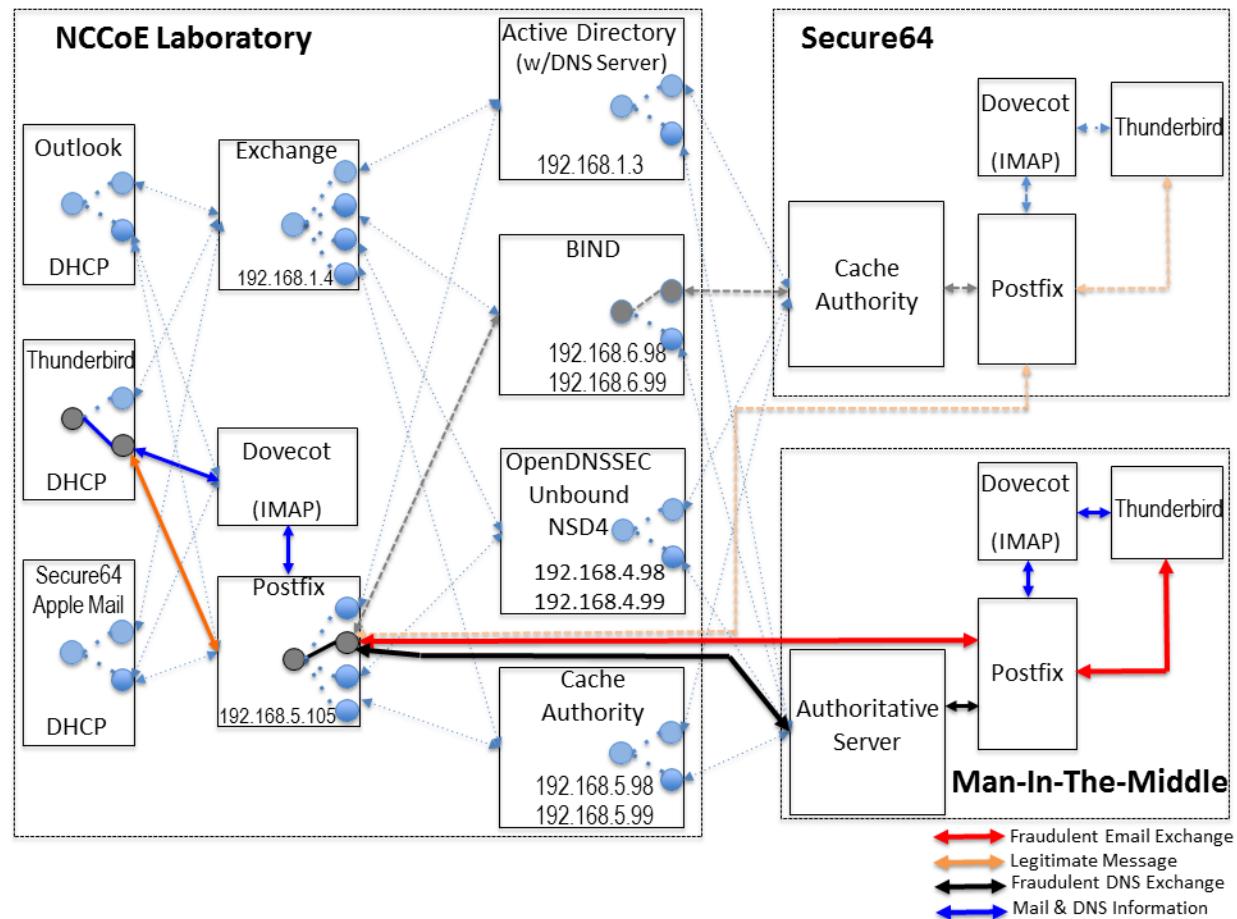


Figure 5.4 Man-In-The-Middle Event Configurations



The following subsections describe the architecture's MUA, MTA, and DNS service components and Cybersecurity Framework Core categories supported by those components.

5.2.1 Client Systems and Mail User Agents (MUAs)

Client systems environments are Microsoft Office, Apple Mail, and open-source Linux-based Thunderbird applications. These include both commercial products and open-source software. MUA capabilities associated with the client systems are used to invoke S/MIME digital signature and signature verification for email, but user-to-user encryption is not demonstrated. Collaborators assisted in installation, integration tailoring as necessary, and testing of laboratory configurations.

Table 5.1 Client Systems

Application	Source	Collaborator Configuration Support	Cybersecurity Framework Category
Office Outlook Mail User Agent	Microsoft	Microsoft	PR.AC-1, PR.AC-2, PR.DS-1, PR.DS-2, PR.DS-6, PR.PT-4, RS.MI-2
Thunderbird Mail User Agent	Open (Mozilla)	NLnet Labs	PR.AC-1, PR.AC-2, PR.DS-1, PR.DS-2, PR.DS-6, PR.PT-4, RS.MI-2
Thunderbird with Apple Key Chain	Secure64	Secure64	PR.AC-1, PR.AC-2, PR.DS-1, PR.DS-2, PR.DS-6, PR.PT-4, RS.MI-2

5.2.2 Email Servers

Email servers include both Windows and Linux-based (Dovecot/Postfix) Mail Transfer Agents. Server-to-server encryption was demonstrated in the Postfix environments. Authentication of domain and server identity was based on DNSSEC-signed DANE records. Use of these DANE records is only supported by Postfix at the time of this project. The MTAs support each of the Cybersecurity Framework Functions, Categories, and Subcategories identified in [section 4.4.4](#) above. The servers were demonstrated in different DNS environments and different TLSA RR usage scenarios. In order to demonstrate representative TLSA parameters, the demonstrations used self-signed certificates, end-entity certificates generated by well-known CAs and end-entities generated by enterprise local CAs.

166

Table 5.2 Mail Transfer Agents

Application	Source	Collaborator Configuration Support	Cybersecurity Framework Category
Exchange 2016 ^a Mail Transfer Agent TLS capable	Microsoft	Microsoft	PR.AC-1, PR.AC-2, PR.AC-3, PR.AC-5, PR.DS-1, PR.DS-239, PR.DS-6, PR.PT-439, DE.CM-1, DE.CM-2, DE.DP-4, RS.RP-1, RS.CO-2, RS.MI-2
Postfix Mail Transfer Agent TLS capable DANE capable	Open (postfix.com)	NLnet Labs Fraunhofer Secure64	PR.AC-1, PR.AC-2, PR.AC-3, PR.AC-5, PR.DS-1, PR.DS-2, PR.DS-6, PR.PT-4, DE.CM-1, DE.CM-2, DE.DP-4, RS.RP-1, RS.CO-2, RS.MI-2

a. Exchange provided integrity protection only for PR.DS-1, PR.DS-2, and PR.PT-4 (Scenario 2).

167

5.2.3 DNS Servers

168

169

170

171

172

173

174

Both Windows and Linux-based DNS server and support components were contributed. DNS services provided include DNSSEC validating DNS resolvers (stub and recursive) and authoritative DNS servers for DNSSEC signed zones. Support for SMIIMEA and TLSA records was demonstrated. The DNS server components support each of the Cybersecurity Framework Functions, Categories, and Subcategories identified in [section 4.4.4](#) above with the exception of PR.DS-1 (protection of data-at-rest).

Table 5.3 DNS Servers

Application	Source	Collaborator Configuration Support	Cybersecurity Framework Category
Active Directory and Windows Server 2016 ■ Supports DNSSEC	Microsoft	Microsoft	PR.AC-1, PR.AC-2, PR.AC-3, PR.AC-5, PR.DS-2, PR.DS-6, PR.PT-4, DE.CM-1, DE.CM-2, DE.DP-4, RS.RP-1, RS.CO-2, RS.MI-2
BIND ^a ■ Supports DNSSEC ■ Supports DANE	Open (ISC)	Internet Systems Consortium (ISC)	PR.AC-1, PR.AC-2, PR.AC-3, PR.AC-5, PR.DS-2, PR.DS-6, PR.PT-4, DE.CM-1, DE.CM-2, DE.DP-4, RS.RP-1, RS.CO-2, RS.MI-2

Table 5.3 DNS Servers

Application	Source	Collaborator Configuration Support	Cybersecurity Framework Category
NSD4 ■ Supports DNSSEC ■ Supports DANE Unbound ■ Supports DNSSEC OpenDNSSEC	Open (NLnet Labs)	Open (NLnet Labs)	PR.AC-1, PR.AC-2, PR.AC-3, PR.AC-5, PR.DS-2, PR.DS-6, PR.PT-4, DE.CM-1, DE.CM-2, DE.DP-4, RS.RP-1, RS.CO-2, RS.MI-2
DNS AUTHORITY DNS MANAGER ■ Supports DNSSEC ■ Supports DANE (Caching authority is labeled DNS CACHE, and signer runs on a dedicated processor)	Secure64	Secure64	PR.AC-1, PR.AC-2, PR.AC-3, PR.AC-5, PR.DS-1, PR.DS-6, PR.PT-4, DE.CM-1, DE.CM-2, DE.DP-4, RS.RP-1, RS.CO-2, RS.MI-2

a. The name BIND stands for “Berkeley Internet Name Domain.”

6 Outcome

This section discusses the security platform from the perspective of the user and the system administrator. We define system administrator as a person within the organization who has elevated privileges on the management systems in the build. System administration functions include identification of system components, system installation, system integration, system configuration, configuration monitoring, identification of exception conditions, system maintenance, and status reporting to management.

6.1 The User's Experience

The user's experience varies from relatively minimal additional impact in enterprise environments with established system administration and support to a significant impact in the case of individual self-supported users. Where the enterprise offers systems administration and support services, the user's experience with respect to DNS services is essentially unchanged. One exception is that, where DNSSEC authentication fails, email messages sent to or by a user will not be delivered. This should be an uncommon experience for correspondents but it is up to the enterprise DNS administrator to prevent this happening.

Similarly, for server-to-server encryption, the security protection features should be essentially transparent to the user.

For user-to-user digital signature, the user must first have a certificate installed in their MUA. This may be included in digital identity credentials, or it may be provided by the system administrator in the process of provisioning the user's computer. Otherwise, the procedure required would be similar to that followed in [section 3.2](#) of SP 1800-6C. The steps required vary from platform to platform (e.g., Windows, Linux, Mac), user agent to user agent (e.g., Outlook vs Thunderbird) and how the private key is stored (on the system, smart cards, etc.). Representative user requirements are described below (in this case for Outlook running on MacBook and Thunderbird running on Linux).

6.1.1 User's Digital Signature Experience with Outlook on MacBook

To use digital signatures and encryption, both the sender and recipient must have a mail application that supports the S/MIME standard (e.g., Outlook).

Note: Before this procedure is started, a certificate must be added to the keychain on the computer. For information about how to request a digital certificate from a certification authority, see MacOS Help or click on "Help" on the Outlook tool bar.

1. On the **Tools** menu, click **Accounts**.
2. Click the account that is to be used to send a digitally signed message, click **Advanced**, and then click the **Security** tab.

35 3. Under **Digital signing**, on the **Certificate** pop-up menu, click the certificate that is to be
36 used.

37 Note: *The Certificate pop-up menu only displays certificates that are valid for digital signing or*
38 *encryption that have already been added to the keychain for the Mac OS X user account. To*
39 *learn more about how to add certificates to a keychain, see Mac OS Help.*

40 4. Do any of the following:

To	Do this
Make sure that the digitally signed messages can be opened by all recipients, even if they do not have an S/MIME mail application and can't verify the certificate	Select the Send digitally signed messages as clear text check box.
Allow the recipients to send encrypted messages to you	Make sure that signing and encryption certificates have been selected on this screen, and then select the Include my certificates in signed messages check box.

42 5. Click **OK**, and then close the **Accounts** dialog box.
43 6. In an e-mail message, on the **Options** tab, click **Security**, and then click **Digitally Sign Message**.
44 7. Finish composing the message, and then click **Send**.

46 6.1.2 User's Digital Signature Experience with Thunderbird

47 For purposes of illustration, the description of the user experience with Thunderbird also
48 included certificate management requirements. The example here shows both S/MIME and
49 PGP examples of certificate management. The S/MIME approach is recommended. Note that
50 when using OpenPGP, a FIPS 140-conformant version should always be used.

51 6.1.2.1 S/MIME Certificate Management

52 S/MIME certificates are used for digitally signed and encrypted e-mail messages. For
53 information about getting or creating S/MIME certificates, see:
54 http://kb.mozilla.org/Getting_an_SMIME_certificate.

55 Installing an S/MIME certificate

56 Note: *Before a user can create or import his or her own certificate and private key, he or she*
57 *must first set a master password if this has not already been done. The master password is*
58 *needed so that imported certificates are stored securely. See*
59 http://kb.mozilla.org/Master_password *for instructions for setting a master password. The*
60 *user may have his or her own personal certificate and private key in a .p12 or .pfx file, and may*
61 *wish to import it into Thunderbird. Once a Master Password has been set, the user can*
62 *import/install a personal S/MIME certificate from a .p12 or .pfx file by doing the following steps.*

- 63 1. Open the Certificate Manager by going to **Tools -> Options... -> Advanced -> Certificates -> Manage Certificates....**
- 64 2. Go to the tab named **Your Certificates**.
- 65 3. Click on **Import**.
- 66 4. Select the **PKCS12** certificate file (.pfx or .p12).
- 67 5. It will ask the user for the master password for the software security device. The user enters his or her master password and clicks **OK**.
- 68 6. Next, it will ask the user for the password protecting his or her personal certificate. If the user's .p12 or .pfx file has a password, he or she enters it here, otherwise leave this field empty. Then click **OK**.

73 The S/MIME certificate should now have been imported. If the certificate was not trusted,
 74 consult the instructions at
 75 http://kb.mozilla.org/Thunderbird_-_FAQs_-_Import_CA_Certificate.

76 Configuring Thunderbird for using the certificate to sign email

77 Go to **Tools -> Account Settings...** in Thunderbird. Then find the account with the email address
 78 that matches the email address in the certificate that has just been installed. Choose **Security**
 79 under that account and select the certificate that has just been installed. The rest of the options
 80 should be self explanatory. When the user selects a certificate in Account Settings, that
 81 selection only applies to the account's default identity or identities. There is no user interface
 82 for specifying certificates for an account's other identities. If desired, this can be worked around
 83 by editing the settings manually, copying the settings from an account's default identity to
 84 some other identity. The settings have names ending in: signing_cert_name, sign_mail,
 85 encryption_cert_name and encryptionpolicy.

86 User Installation of a Self-Signed S/MIME Certificate

87 If the SMIME certificate in a user's .p12 or .pfx file is a self-signed certificate for the user's own
 88 identity, then before that file can be installed into the tab named **Your Certificates**, the user
 89 must first install that certificate as a certificate authority in the **Authorities** tab. The PKCS12
 90 certificate file will not install into the **Authorities** tab. The user will need a copy of a self-signed
 91 certificate that does not contain the user's private key. This is usually in the form of a .cer file.
 92 One way to obtain the .cer form of a certificate from the .p12 file is to use the Firefox Add-on
 93 Key Manager to extract the .cer certificate from the .p12 file. With that Add-on installed in
 94 Thunderbird, the user goes to **Tools -> Key Manager Toolbox -> Key Manager -> Your Keys**,
 95 select his or her key, selects **Export** and chooses **X.509** as file format.

- 96 1. Go to **Tools -> Options... -> Advanced -> Certificates -> Manage Certificates....**
- 97 2. Go to the **Authorities** tab.
- 98 3. Click on **Import**.
- 99 4. Select the **.cer** file.
- 100 5. It will ask the user for what purposes he or she wants to trust the certificate. Select **Trust this CA to identify email users**.

102 6. Click **OK** to complete the import.

103 Note: *Thunderbird automatically adds other people's S/MIME certificates to the **Other People's***
 104 *tab of a user's Certificate Manager when he or she receives from them a digitally signed*
 105 *message with a valid signature and with an S/MIME certificate issued by a recognized and*
 106 *trusted Certificate Authority (CA). CA certificates that appear in Thunderbird's "Authorities" tab*
 107 *are recognized, and may also be trusted. CA certificates that do not appear in that tab are*
 108 *considered **unrecognized**. An S/MIME certificate that was issued by an unrecognized CA will*
 109 *not be automatically added to the **Other People's** tab of the user's Certificate Manager. If the*
 110 *user attempts to manually import an S/MIME certificate that was issued by an unrecognized CA,*
 111 *nothing will happen--literally. Thunderbird will not even display an error dialog. It will just not*
 112 *import the S/MIME certificate. This is generally not a problem when receiving an S/MIME*
 113 *certificate that was issued by a trusted Certificate Authority (CA), but could be a problem for a*
 114 *certificate that was issued by an unrecognized or untrusted CA, or for a certificate that is*
 115 *self-signed (i.e. it has no CA other than itself). So, before a user can import an S/MIME*
 116 *certificate that is issued by an unrecognized CA or is self-signed, he or she must first acquire*
 117 *and import the certificate for the issuing CA. In the case of a self-signed certificate, a .cer file*
 118 *needs to be acquired from the individual whose certificate the user wishes to add.*

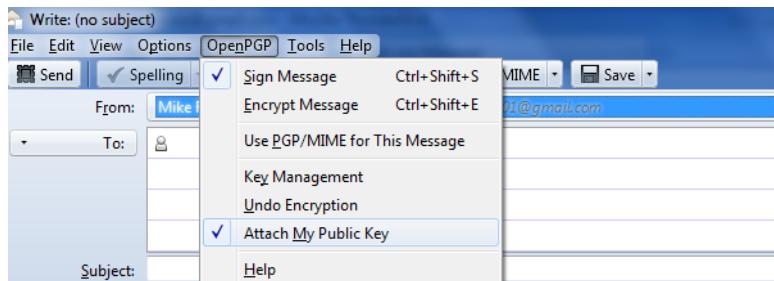
119 **6.1.2.2 PGP Example of Sending and Receiving Public Keys**

120 **Sending a public key via email**

121 To send signed messages to other people, the user must first send them the public key:

122 1. Compose the message.

123 2. Select **OpenPGP** from the Thunderbird menu bar and select **Attach My Public Key**.



124 3. Send the email as usual.

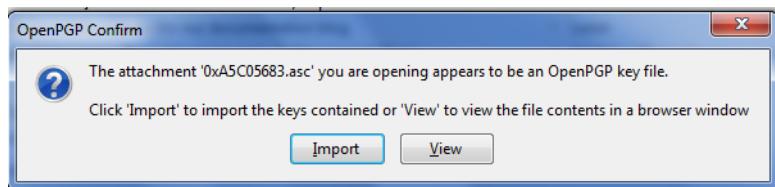
125 **Receiving a public key via email**

126 To verify signed messages from other people, the public key must be received and stored:

127 1. Open the message that contains the public key.

128 2. At the bottom of the window, double click on the attachment that ends in .asc. (This file
 129 contains the public key.)

131 3. Thunderbird automatically recognizes that this is a PGP key. A dialog box appears, 132 prompting the **Import** or **View** of the key. Click **Import** to import the key.

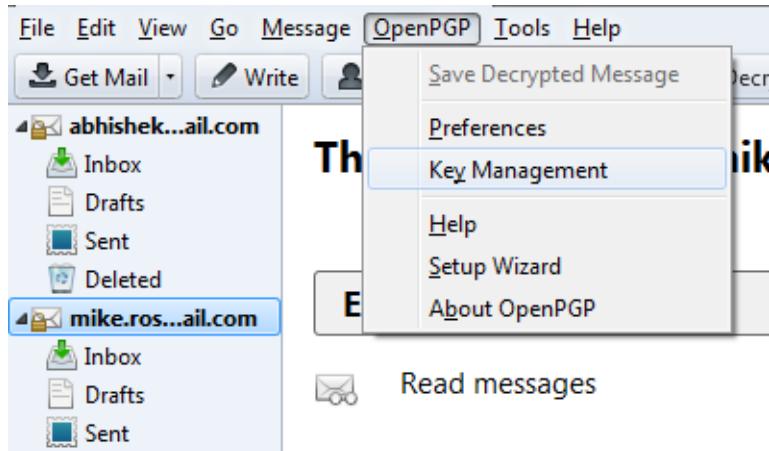


133 134 4. A confirmation that the key has been successfully imported will be shown. Click **OK** to 135 complete the process.

136 Revoking a key

137 If the private key may have been compromised (that is, someone else has had access to the file 138 that contains the private key), revoke the current set of keys as soon as possible and create a 139 new pair. To revoke the current set of keys:

140 1. On the Thunderbird menu, click **OpenPGP** and select **Key Management**.



141 142 2. A dialog box appears as shown below. Check **Display All Keys by Default** to show all the 143 keys.

144 3. Right-click on the key to be revoked and select **Revoke Key**.

145 4. A dialog box appears asking the user if he or she really wants to revoke the key. The user 146 clicks **Revoke Key** to proceed.

147 5. Another dialog box appears asking for the entry of a secret passphrase. The user enters the 148 passphrase and clicks **OK** to revoke the key.

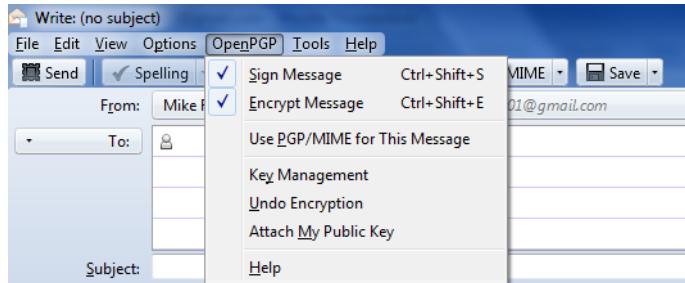
149 6. The user sends the revocation certificate to the people with whom he or she corresponds 150 so that they know that the current key is no longer valid. This ensures that if someone tries 151 to use the current key to impersonate the user, the recipients will know that the key pair is 152 not valid.

153 **6.1.2.3 Sending a Digitally Signed Email**

154 1. Compose the message as usual.

155 2. To digitally sign a message, select **OpenPGP** from the Thunderbird menu and enable the

156 **Sign Message** option.

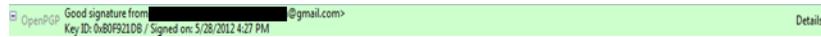


158 3. If the email address is associated with a cryptographic certificate, the message will be
159 signed with the key contained in that certificate. If the email address is not associated with
160 a cryptographic certificate, a certificate must be selected from a list.

161 4. Send the message as usual.

162 **6.1.2.4 Reading a Digitally Signed Email**

163 When a signed message is received, and If Thunderbird recognizes the signature, a green bar
164 (as shown below) appears above the message. To determine whether or not the incoming
165 message has been signed, look at the information bar above the message body.¹



167 If the message has been signed, the green bar also displays the text, "Signed message". A
168 message that has not been signed could be from someone trying to impersonate someone else.

169 **6.2 The System Administrator's Experience**

The system administrator(s) will generally be responsible for configuring the MUAs, MTA, and DNS servers. Specific installation and configuration instructions and examples are provided in Sections 2, Section 3, Appendix F, Appendix G, and Appendix H of the [How-To Guides](#), SP 1800-6C. Configuration includes setting up and publishing certificates in the DNS as TLSA and SMIMEA RRs. Certificate management using Well-Known CA-issued certificates or Enterprise CA-issued certificates is required for federal government applications and is strongly recommended in other applications. While instructions for configuration for DNSSEC are provided for environments described in SP 1800-6C, this more secure set of configuration options are not generally invoked by default. Therefore, more effort and expertise are needed on the part of the DNS administrator.

1. If the message is also encrypted on a user-to-user basis, Thunderbird will also ask for the entry of a secret passphrase to decrypt the message.

180 Configuring and activation of mail servers (MTAs) for channel encryption by default is described
181 in section 3.3 of SP 1800-6C. Summary information is provided here and in links for illustration
182 purposes for Microsoft Office 365 Exchange and Postfix.

183 In general, the bulk of the system administrator's effort is in acquiring and publishing the
184 necessary certificates. Maintenance of the security functions, once they've been set up, is a
185 relatively routine system administration activity.

186 6.2.1 Microsoft Exchange

187 Only Microsoft Exchange for Office 365 encrypts users' data while it's on Microsoft servers and
188 while it's being transmitted between the MTSSs. Exchange for Office 365 does provide controls
189 for end users and administrators to fine tune what kind of encryption is desired to protect files
190 and email communications.

191 6.2.2 Postfix

192 Postfix TLS support is described at http://www.postfix.org/TLS_README.html. Postfix can be
193 configured to always use TLS when offered by receivers.²

2. "Setting Postfix to encrypt all traffic when talking to other mail servers," *Snapdragon Tech Blog*, August 9, 2013. <http://blog.snapdragon.cc/2013/07/07/setting-postfix-to-encrypt-all-traffic-when-talking-to-other-mailservers/>

7 Evaluation

2	7.1 Assumptions and Limitations	54
3	7.2 Testing.....	54
4	7.3 Scenarios and Findings.....	57

5

7.1 Assumptions and Limitations

This security characteristic evaluation has the following limitations:

- It is not a comprehensive test of all security components, nor is it a red team exercise.
- It cannot identify all weaknesses.
- It does not include the lab infrastructure. It is assumed that its devices are hardened. Testing these devices would reveal only weaknesses in implementation that would not be relevant to those adopting this reference architecture.

7.2 Testing

The evaluation included analysis of the security platforms to identify weaknesses and to discuss mitigations. The focus of this portion of the evaluation was hands-on testing of the laboratory build and examination of product manuals and documentation. Our objective was to evaluate the building block and not specific products. The presence of four primary OSs for domains tested (Linux, MacOS, SourceT Micro OS, and Windows) made complete product-independent hands-on testing unrealistic.

Table 7.1 describes the goals of each sequence of test cases. For each sequence, the Cybersecurity Framework (CSF) Subcategories and associated SP 800-53 control(s), the test environment(s) involved, and evaluation objective of the test are identified. The results of the tests are provided NIST SP 1800-6c.

In all test sequences the sending MTA attempted to establish a TLS protected channel to deliver the email message to the receiver. In the attack scenarios, a malicious actor attempts to disrupt this transfer. In all test sequences, the sending MUA signed the message, and the receiving MUA, checked the signature. Exchange was used only for Scenario 2.¹ In all test sequences, the sending MTA attempted to verify the correctness of all DNS responses via DNSSEC validation. In most scenarios, alice@<somedomain> sent an email to bob@<receivername>. Both senders and receivers had their own (separate) DNS infrastructures consisting of both authoritative and recursive servers. The Exchange as Sender tests were conducted for completeness and for examples of SMTP over TLS w/o DANE support - what it looked like and how well it worked.

1. Exchange MTAs did not attempt to encrypt or decrypt MTA-to-MTA message exchanges.

³³ **Table 7.1** **Tests Performed**

Test Sequence	CSF Subcategories	SP 800-53 Controls	Configuration	Evaluation Objective
Sequence 1	PR.AC-1 PR.AC-2 PR.DS-1 PR.DS-2 PR.DS-6 RS.MI-2	AC-2, AC-17, AC-19, AC-20, IA Family, IR-4, SC-8, SC-28, SI-7	<p>An Outlook MUA, interfacing with an Exchange MTA, was configured to use Active Directory and BIND DNS services in turn. Each of the six configurations exchanged email with</p> <ul style="list-style-type: none"> ■ a Secure64 MUA/MTA/DNS service stack that included a Postfix MTA and a Thunderbird MUA running on a Mac OS system ■ an NLnet Labs MUA/MTA/ DNS service stack that included a Postfix MTA and a Thunderbird MUA running on Linux <p>The events include events showing use of Well-Known CAs (CU-1), Enterprise CAs (CU=2), and Self-Signed Certificates (CU=3) for TLS and S/MIME-enabled mail receivers and S/MIME. Figure 5.2 above depicts the set-up for laboratory support for the Secure64 destination variant of this test sequence.^a</p>	Email messages between Postfix MTAs were encrypted and successfully decrypted via TLS. (Scenario 1). Signature was logged. All messages were S/MIME signed. Outlook attempted to verify received messages (Scenario 2). Signature verification results were noted. DNS name verification results were noted.
Sequence 2	PR.AC-1 PR.AC-2 PR.DS-1 PR.DS-2 PR.DS-6 RS.MI-2	AC-2, AC-17, AC-19, AC-20, IA Family, IR-4, SC-8, SC-28, SI-7	<p>Outlook and Thunderbird MUAs, configured to use a Postfix MTA with Dovecot IMAP support, were configured in turn to use BIND and Secure64's DNS Authority, DNS Cache, and DNS Signer implementations. Each of the six configurations exchanged email with a Secure64 MUA/MTA/ DNS service stack that included a Thunderbird MUA, Postfix/Dovecot MTA, and DNS Signer/DNS Cache/DNS Authority services for processing received messages; and an NLnet Labs MUA/MTA/ DNS service stack that included a Thunderbird MUA, Postfix/Dovecot MTA, and NSD4, Unbound, and OpenDNSSEC DNS services. The test events include using Well-Known CA issued (TLSA/SMIMEA CU=1), Enterprise CA issued (CU=2), and Self-Signed Certificates (CU=3). Figure 5.2 above depicts the set-up for laboratory support for this test sequence.</p>	Email messages between MTAs were encrypted and successfully decrypted. (Scenario 1). Signature and encryption were logged. All messages were S/MIME signed. Outlook attempted to verify received messages (Scenario 2). Signature verification results were noted. DNS name verification results were noted.

Table 7.1 Tests Performed

Test Sequence	CSF Subcategories	SP 800-53 Controls	Configuration	Evaluation Objective
Sequence 3	PR.AC-1 PR.AC-2 PR.AC-3 PR.AC-5 PR.DS-2 RS.MI-1	AC-2, AC-4, AC-17, AC-19, AC-20, IA Family, IR-4, SC-7, SC-8	Fraudulently S/MIME-signed email was sent from a malicious sender to recipients using Outlook and Thunderbird MUAs configured to use Exchange and Postfix as MTAs. The Outlook/Exchange configuration used Active Directory as its DNS server. The configurations employing Postfix/Dovecot MTAs were demonstrated with each of the other three contributed DNS Services. In one event, the Thunderbird MUA employed an Apple Key Chain Utility tool that allows a host to obtain X.509 certificates via of DANE RRs. All events were conducted using well-known CA and Enterprise CA-issued certificates for the impersonated sender. The set-up for this sequence is depicted in Figure 5.3 above.	The fraudulent site attempted to spoof a valid sending domain belonging to a Secure64 site. An Outlook/Exchange/ Active Directory set-up acted as the fraudulent site. The email exchange between organizations was carried over TLS, and the email message was S/MIME signed on the fraudulent users' client device. Where Well-Known CA-issued certificates or Enterprise CA-issued certificates were used, and the MTA was DANE aware. The MUA using a SMIMEA utility was able to detect the fraudulent email and mark the email as not validated.
Sequence 4	PR.AC-1 PR.AC-2 PR.AC-3 PR.AC-5 PR.DS-2 PR.DS-6 RS.MI-1 RS.MI-2	AC-2, AC-4, AC-17, AC-19, AC-20, IA Family, IR-4, SC-7, SC-8, SI-7	The sender used an Outlook MUA sending mail through a Postfix/Dovecot MTA and using (in turn): Active Directory and DNS Server, BIND DNS Server, and NLnet Labs DNS Services. Self-signed certificates were used on the legitimate receiver side (TLSA RR parameter CU=3) for TLS. Each of the three configurations attempted to initiate an email exchange with an external Secure64 site. The set-up for this sequence is depicted in Figure 5.4 above.	The man-in-the-middle, an Outlook/Exchange/Active Directory stack, attempted to intercept the email from the NCCoE Laboratory Configuration by acting as a Man-in-the-Middle. The email and DNS transactions were logged in each case, and the results are provided in Volume C Appendix C. Where the MTA was DANE-aware, A detected spoofing. The mail connection to the MTA was established but closed the connection before the mail was transferred. Otherwise, the MTA failed to detect the man-in-the-middle and sent the email.

Table 7.1 Tests Performed

Test Sequence	CSF Subcategories	SP 800-53 Controls	Configuration	Evaluation Objective
Sequence 5	PR.AC-1 PR.DS-6 PR.CM-1 PR.DP-4 PR.CO-2	AC-4, IR-5, SC-5, SC-20, SC-21, SC-23, SI-4, SI-13	A DANE-enabled Postfix MTA sent message traffic to four MTAs with one Authoritative Server serving all four zones. An NSD4 Authoritative DNS server and Unbound recursive server were provided for the Postfix sending MTA, and a Secure64 DNS Authority and Signer provided the DNS services for the recipient zones. We reviewed the log files. One of the recipient MTAs did not employ TLSA, one employed a valid TLSA with the CU set to 3, one employed a TLSA with a certificate usage field of 1, but with an incomplete (i.e. bad) PKI certification path (PKIX failure), and one employed mismatched server cert/TLSA with the certificate usage field set to 3 (DANE validation failure).	A large number of email messages are generated in the Postfix server device using a Python script, and the Postfix MTA sends the messages to each of four recipient MTAs in different zones. In the recipient MTA running without TLSA and that running with a valid matching TLSA and certificate usage field set to 3, all messages should be accepted. In the recipient MTA with a TLSA RR using certificate usage of 1, but with an incomplete PKIX validation path, and the recipient MTA with a mismatched certificate/TLSA (cert usage 3), the sender should close the connection without sending the message. Logwatch running on the sending Postfix server device logged the instances of failure to deliver due to certificate expiration or bad certificate path.

a. The connections depicted in the Figure are actually for the Secure64 variant of the first Sequence 2 configuration. Capabilities for Sequence 1 support are shown as dotted lines.

³⁴ 7.3 Scenarios and Findings

³⁵ One aspect of our security evaluation involved assessing how well the reference design addresses the objectives of the scenario it was intended to support.

7.3.1 Scenario 1

38 Scenario 1 involved the ordinary exchange of email between two organizations' email servers
 39 carried over TLS, where the TLS key management was protected by DANE and DNSSEC. Private
 40 certificates were generated by either well-known CAs, enterprise local CAs or self-signed. User
 41 connections to their organizations' respective mail servers were established and maintained
 42 within a physically protected zone, and email was encrypted between mail servers using TLS.
 43 The confidentiality of encryption keys was maintained such that no unauthorized third party
 44 had access to the keys. The mail servers used X.509 certificates to store and transport public
 45 keys to establish the TLS channel. DNSSEC ensured that each sending mail server receives the IP
 46 address to the legitimate and authorized receiving mail server and (if applicable) validate its
 47 X.509 certificate. DANE bound the cryptographic keying material to the appropriate server. TLS
 48 was used to protect the confidentiality of the email exchange. Encryption of the email message
 49 was accomplished by the originator's email server, and decryption of the email message was
 50 accomplished by the recipient's email server using standard server libraries.

51 The tests included an attempt by a fraudulent mail server to pose as the legitimate mail
 52 receiver for a domain. The tests also include a man-in-the-middle attack to attempt to disrupt
 53 the TLS connection with the objective of achieving an unencrypted transmission of the email.
 54 Both attempts failed due to use of DNSSEC and DANE. In both cases, an indication was made
 55 available to the sending email server when the DNSSEC signature associated with the domain
 56 data is determined to be invalid.

7.3.2 Scenario 2

57 Scenario 2 involved end-to-end signed email, where the email exchanges between
 58 organizations were carried over TLS as in (1), the email messages were signed and verified with
 59 S/MIME on the end-users' client devices, and the S/MIME key management was protected by
 60 DANE and DNSSEC. Private certificates were generated by well-known and enterprise local CAs.
 61 Self-signed certificates were not used. Individuals established connections to their domains'
 62 respective mail servers within a physically protected zone of control. Cryptographic digital
 63 signatures were applied to messages to provide authentication and integrity protection for the
 64 email. S/MIME was the protocol used for the digital signing. These certificates were then
 65 encoded in the DNS using the appropriate DANE DNS record type. DNSSEC ensured that each
 66 originating user's mail server connects to the intended recipient's mail server. DANE bound the
 67 cryptographic keying material to the appropriate server and individual user digital signature
 68 certificates. TLS was employed to protect the confidentiality of the email. Digital signing of
 69 email messages was accomplished by originator's MUA, and checking the validity of the
 70 signature (hence the integrity of the authorization provided in the email message) was
 71 accomplished by recipient's MUA.

72 The tests in this scenario included an attempt by a fraudulent actor to pose as an originator of
 73 the email. This attempt failed due to use of DNSSEC and DANE. The receiving MUA, using a third
 74 party SMIMEA tool, was able to fetch the senders real S/MIME certificate from the DNS and
 75 confirm that the fraudulent email was signed using a different certificate.

7.3.3 Effects of DANE Errors

In addition to the scenarios described above, a DANE-enabled Postfix MTA sent message traffic to four other postfix MTAs. A single BIND instance was set up to serve the TLSA and A RRs for the four receivers. One of the receiving MTAs did not employ DANE. The second employed DANE with a valid TLSA with the certificate usage field² set to 3. The third employed a TLSA with a certificate usage field of 2, but with an incomplete (i.e. bad) PKI certification path (generating a PKIX validation failure). The TLSA contained a local enterprise trust anchor, but the server did not have the full certificate chain (missing intermediate certificate). The final one employed DANE with a TLSA RR using Certificate Usage of 3, but there was a mismatch between the server cert and TLSA RR (generating a DANE validation failure).

Little or nothing appeared in the sender's logs for messages sent to either the MTA not employing TLS or the employing a valid TLSA. The growth rates for logs for the MTA that employed a TLSA with a certificate usage field of 1, but with a PKIX failure and the one that employed mismatched server cert/TLSA (i.e. DANE validation failure) were measured.

When the sender was configured to never use TLS, the mail was sent in plaintext regardless of the TLS/DANE configuration of the receiver. When the sender was configured to use TLS opportunistically, it used TLS regardless of the status of the certificate, or TLSA. In fact, the sender did not issue a query to find TLSA RRs even if published. When the sender used opportunistic DANE, it used TLS when available regardless of the DANE validations results. If validation failed, the mail was still sent and the result was logged as an "Untrusted" or "Anonymous" TLS connection, depending on the presence of a TLSA RR.

Of the four options used in the lab, *dane-only* is the most rigorous in what a sender would accept before sending mail. When the receiver did not offer the STARTTLS option, or lacked a TLSA RR, mail was not sent. Likewise, if a TLSA RR was present, but there was an error in validation (either the TLSA RR itself had an error, or PKIX failed), the mail was not sent. Therefore, use of this option is not recommended for general use as this will result in the majority of email being deferred. It should only be used in scenarios where senders and receivers are coordinated and maintain a stable DANE deployment.

2. RFC 6698, *The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA*, Section 2.1.1. <https://tools.ietf.org/html/rfc6698#section-2.1.1>

8 Future Build Considerations

Both public sector and private sector enterprises are heavily dependent on web-based technology other than email for e-commerce and other public-facing applications. Fraudulent web sites pose at least as great a security and privacy problem as fraudulent email. Further, as email becomes a more difficult medium for malicious entities to use as a penetration vector, other web-based media will be more intensively exploited. Already, emerging communications trends appear to be replacing email exchanges among individuals with other social media (e.g., Baidu, Facebook, Facebook Messenger, Google+, Instagram, Linkedin, Pinterest, Snapchat, Tieba, Tumblr, Twitter, Viber, WhatsApp, and YouTube). Therefore, an extension of the current project that focuses on use of improved DNSSEC applications such as DANE for web applications other than mail may be justified.

Additionally, the test scenarios did not include the Exchange for Office 365 MTA to demonstrate Scenario 1. Future builds might be considered to demonstrate this capability.

Finally, utilities are currently under development that would provide improved support for SMIMEA and improved system notification of failed DNSSEC signature validation events. Future builds might be considered to demonstrate these capabilities as well.

Appendix A Acronyms

2	ASN	Abstract Syntax Notation
3	AXFR	DNS Full Zone Transfer Query Type
4	BIND	Berkeley Internet Name Daemon
5	BSD	Berkeley Software Distribution
6	CA	Certificate Authority
7	CKMS	Cryptographic Key Management System
8	CRL	Certificate Revocation List
9	CU	Certificate Usage Type
10	DANE	DNS-based Authentication of Named Entities
11	DNS	Domain Name System
12	DNSSEC	DNS Security Extensions
13	Email	Electronic Mail
14	EMC	Electromagnetic Compatibility
15	EMI	Electromagnetic Interference
16	FCKMS	Federal Cryptographic Key Management System
17	FIPS	Federal Information Processing Standard
18	HIPAA	Health Insurance Portability and Accountability Act
19	IEC	International Electrotechnical Commission
20	IEEE	Institute of Electrical and Electronics Engineers
21	IETF	Internet Engineering Task Force
22	IP	Internet Protocol
23	IRS	Internal Revenue Service
24	ISO	Internet Organization for Standardization
25	ITL	Information Technology Laboratory
26	MIME	Multipurpose Internet Mail Extension
27	MTA	Mail Transfer Agent
28	MUA	Mail User Agent
29	MX	Mail Exchange (Resource Record)
30	NCCoE	National Cybersecurity Center of Excellence
31	NIST	National Institute of Standards and Technology
32	OS	Operating System
33	PKI	Public Key Infrastructure

34	PKIX	Public Key Infrastructure X.509
35	RFC	Request for Comments
36	RMF	Risk Management Framework
37	RR	Resource Record
38	S/MIME	Secure/Multipurpose Internet Mail Extensions
39	SMIMEA	S/MIME Certificate Association (Resource Record)
40	SMTP	Simple Mail Transfer Protocol
41	SP	Special Publication
42	SQL	Structured Query Language
43	TLS	Transport Layer Security
44	TLSA	TLS Certificate Association (Resource Record)
45	UA	User Agent
46	VLAN	Virtual Local Area Network
47	VM	Virtual Machine

Appendix B References

2 *Securing the Federal Government's Domain Name System Infrastructure*, Executive Office of the
3 President, Office of Management and Budget, M-08-23, August 22, 2008. <https://www.whitehouse.gov/sites/default/files/omb/memoranda/fy2008/m08-23.pdf>

5 *Enhancing the Security of Federal Information and Information Systems*, Executive Office of the
6 President, Office of Management and Budget, M-14-03, November 18, 2013. <http://www.whitehouse.gov/sites/default/files/omb/memoranda/2014/m-14-03.pdf>

8 *Improving Critical Infrastructure Cybersecurity*, Executive Office of the President, Executive
9 Order 13636, February 12, 2013. <https://www.federalregister.gov/articles/2013/02/19/2013-03915/improving-critical-infrastructure-cybersecurity>

11 *Federal Information Security Management Act*, United States Congress, Public Law 107-347,
12 December 17, 2002. <https://www.govtrack.us/congress/bills/107/hr2458>

13 *Gramm-Leach-Bililey Act*, United States Congress, Public Law 104-191, August 21, 1996. <https://www.gpo.gov/fdsys/pkg/PLAW-106publ102/html/PLAW-106publ102.htm>

15 *Health Insurance Portability and Accountability Act*, United States Congress, Public Law 106-
16 102, November 12, 1999. <https://aspe.hhs.gov/report/health-insurance-portability-and-accountability-act-1996>

18 *Managing Information as a Strategic Resource*, OMB Circular A-130, Executive Office of the
19 President, Office of Management and Budget, July 28, 2016. <https://www.federalregister.gov/documents/2016/07/28/2016-17872/revision-of-omb-circular-no-a-130-managing-information-as-a-strategic-resource>

22 *Rules Governing Practice before the Internal Revenue Service*, Internal Revenue Service, Circular
23 Number 230, Revised June 2014. <https://www.irs.gov/tax-professionals/circular-230-tax-professionals>

25 *Security Requirements for Cryptographic Modules*, Federal Information Processing Standard
26 (FIPS), FIPS 140-2, May 2001 (including change notices as of 12-03-2002). <http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf>

28 *Guide for Conducting Risk Assessments*, NIST Special Publication, SP 800-30 Revision 1, Joint
29 Transformation Initiative, September 2012. http://csrc.nist.gov/publications/nistpubs/800-30-rev1/sp800_30_r1.pdf

31 *Guide for Applying the Risk Management Framework to Federal Information Systems: A security
32 Lifecycle Approach*, NIST Special Publication, SP 800-37 Rev. 1, Joint Task Force
33 Transformation Initiative; February 2010 with updates as of June 5, 2014. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-37r1.pdf>

35 *Guidelines on Electronic Mail Security*; NIST Special Publication; SP 800-45 Ver. 2; Tracy, Jansen,
36 Scarfone, Butterfield; February 2007. <http://csrc.nist.gov/publications/nistpubs/800-45-version2/SP800-45v2.pdf>

38 *Federal S/MIME V3 Client Profile*, NIST Special Publication, SP 800-49, Chernick, November
39 2002. <http://csrc.nist.gov/publications/nistpubs/800-49/sp800-49.pdf>

40 *Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS)*
 41 *Implementations*; NIST Special Publication; SP 800-52 Rev. 1; Polk, McKay, Chokhani;
 42 April 2014. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-52r1.pdf>

43 *Security and Privacy Controls For Federal Information Systems And Organizations*, NIST Special
 44 Publication, SP 800-53 Rev. 4, Joint Task Force Transformation Initiative, April 2013.
 45 <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf>

46 *Recommendation for Key Management: Part 1 - General*, NIST Special Publication 800-57 Part
 47 Rev.4, Barker, January 2016. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf>

48 *Recommendation for Key Management: Part 2 - Best Practices for Key Management
 49 Organization*, NIST Special Publication 800-57 Part 2, Barker, Barker, Burr, Polk, and
 50 Smid, August 2005. <http://csrc.nist.gov/publications/nistpubs/800-57/SP800-57-Part2.pdf>

51 *Recommendation for Key Management: Part 3: Application-Specific Key Management
 52 Guidance*, NIST Special Publication, SP 800-57 Part 3 Rev. 1, Barker and Dang, January
 53 2015. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57Pt3r1.pdf>

54 *Electronic Authentication Guideline*; SP 800-63-2; Burr, Dodson, Newton, Perlner, Polk, Gupta,
 55 Nabbus; August 2013. doi:10.6028/NIST.SP.800-63-2 [Direct Link]

56 *Secure Domain Name System (DNS) Deployment Guide*, NIST Special Publication, SP 800-81-2,
 57 Chandramouli and Rose, September 2013. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-81-2.pdf>

58 *A Framework for Designing Cryptographic Key Management Systems*; NIST Special Publication;
 59 SP 800-130; Barker, Branstad, Smid, Chokhani; August 2013. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-130.pdf>

60 *A Profile for U.S. Federal Cryptographic Key Management Systems (CKMS)*; Third Draft; NIST
 61 Special Publication; SP 800-152; Barker, Smid, Branstad; December 18, 2014. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-152.pdf>

62 *Systems Security Engineering: An Integrated Approach to Building Trustworthy Resilient
 63 Systems*, Draft, NIST Special Publication, SP 800-160, Ross, McEvilley, Oren, May 2016.
 64 http://csrc.nist.gov/publications/drafts/800-160/sp800_160_second-draft.pdf

65 *Trustworthy Email*; NIST Special Publication; DRAFT SP 800-177; Chandramouli, Garfinkle,
 66 Nightingale and Rose; Draft Publication; March 29, September 2016. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-177.pdf>

67 "Internet of Things: Standards and Guidance from the IETF", *IETF Journal*, Keränen and
 68 Bormann, April 2016. <https://www.internetsociety.org/publications/ietf-journal-april-2016/internet-things-standards-and-guidance-ietf>

69 *X.509 Certificate Policy for the U.S. Federal PKI Common Policy Framework*, Version 1.21. <http://www.idmanagement.gov/documents/common-policy-framework- certificate-policy>

70 *Internet Protocol*, RFC 791, DARPA, September 1981. <https://tools.ietf.org/html/rfc791>

71 *Domain Names - Concepts And Facilities*, RFC 1034, Mockapetris, November 1987. <https://www.ietf.org/rfc/rfc1034.txt>

72 *Domain Name System Structure and Delegation*, RFC 1591, Postel, March 1994. <https://tools.ietf.org/html/rfc1591>

Internet X.509 Public Key Infrastructure Certificate and CRL Profile; IETF RFC 2459; Housley, Ford, Polk, Solo; January 1999. <https://www.rfc-editor.org/rfc/rfc2459.txt>

The Secure HyperText Transfer Protocol, RFC 2660, Rescorla and Schiffman, August 1999. <https://tools.ietf.org/html/rfc2660>

Threat Analysis of the Domain Name System (DNS), IETF RFC 3833, Atkins and Austein, August 2004. <https://tools.ietf.org/html/rfc3833>

A Method for Storing IPsec Keying Material in DNS, RFC 4025, Richardson, February 2005.
<https://tools.ietf.org/html/rfc4025>

DNS Security Introduction and Requirements; RFC 4033; Arends, Austein, Larson, Massey, and Rose; March 2005. <https://www.ietf.org/rfc/rfc4033.txt>

A Border Gateway Protocol 4 (BGP-4); RFC 4271; Rekhter, Li, and Hares; January 2006. <https://tools.ietf.org/html/rfc4271>

The Transport Layer Security (TLS) Protocol Version 1.2, RFC 5246, Dierks and Rescorla, August, 2008. <https://tools.ietf.org/html/rfc5246>

Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile; Proposed Standard; IETF RFC 5280; Cooper, Santesson, Farrell, Boeyen (Entrust), Housley, Polk; May 2008. <https://datatracker.ietf.org/doc/rfc5280/>

Simple Mail Transfer Protocol, IETF RFC 5321, Draft Standard, Kleinsteiner, October 2008. <https://tools.ietf.org/html/rfc5321>

Secure/Multipurpose Internet Mail Extensions (S/MIME), Version 3.2, Message Specification, Proposed Standard, IETF RFC 5751, ISSN: 2070-1721, Ramsdell and Turner, January 2010. <https://tools.ietf.org/html/rfc5751>

Use Cases and Requirements for DNS-Based Authentication of Named Entities (DANE), IETF RFC 6394, ISSN: 2070-1721, Barnes, October 2011. <https://tools.ietf.org/html/rfc6394>

The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security Protocol: TLSA, Proposed Standard, IETF RFC 6698, ISSN: 2070-1721, Hoffman and Schlyter, August 2012. <https://tools.ietf.org/html/rfc6698>

DNS-Based Service Discovery, RFC 6763, Cheshire and Krotchmal, February 2013. <https://tools.ietf.org/html/rfc6763>

Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, Proposed Standard, IETF RFC 6818, ISSN: 2070-1721, Yee, January 2013. <https://tools.ietf.org/html/rfc6818>

SMTP security via opportunistic DANE TLS, RFC 7672, Dukhovni and Hardaker, May 26, 2015.
<https://tools.ietf.org/html/rfc7672>

Using Secure DNS to Associate Certificates with Domain Names For S/MIME, IETF Internet Draft, draft-ietf-dane-smime-09, Hoffman and Schlyter, September 3, 2015. <https://tools.ietf.org/html/draft-ietf-dane-smime-09>

Domain Name System-Based Security for Electronic Mail, Barker, National Institute of Standards and Technology's Dakota Consulting IDIQ Contract SB1341-12-CQ-0011, Task Order 15-421 Task 3 Report #2, December 17, 2016. https://nccoe.nist.gov/sites/default/files/library/NCCoE_DNS-Based_Secure_E-Mail_BB.pdf

124 Task 2: Report #1 on Standards Review and Support for NCCoE Project Activities, Barker,
125 National Institute of Standards and Technology's Dakota Consulting IDIQ Contract
126 SB1341-12-CQ-0011, Task Order 15-421 Task 2 Report #1, November 30, 2015.

127 Task 3: Report #1 on Standards Review and Support for NCCoE Project Activities, Barker,
128 National Institute of Standards and Technology's Dakota Consulting IDIQ Contract
129 SB1341-12-CQ-0011, Task Order 15-421 Task 3 Report #1, November 30, 2015.

Appendix C DNS-Based Email Security Project Mapping to the Framework Core and Informative References

The following tables map informative NIST and consensus security references to Framework Core subcategories that are addressed by the DNS-Based Email Security platform set. The references do not include protocol specifications that are implemented by the individual products that comprise the demonstrated security platforms. While some of the references provide general guidance that informs implementation of referenced Framework Core functions, the NIST Special Publication references provide specific recommendations that should be considered when composing and configuring security platforms from DNS and email components, implement DNSSEC and mail security platforms, and operating email systems securely.

⁹ **Table C.1 PROTECT (PR)**

Category	Subcategory	Informative References
Access Control (PR.AC): Access to assets and associated facilities is limited to authorized users, processes, or devices, and to authorized activities and transactions.	PR.AC-1: Identities and credentials are managed for authorized devices and users	<p>NIST SP 800-45 Ver. 2 3, 6</p> <p>NIST SP 800-53 Rev. 4 AC-2, IA Family</p> <p>NIST SP 800-57 Part 2 3.1.2.1.3, A.3.2, B.5</p> <p>NIST SP 800-81-2 11.7.2</p> <p>NIST SP 800-130 2.1, 5, 6.4.2, 6.4.23, 6.5, 6.6.1, 6.6.2, 6.7.1, 8.2.4</p> <p>NIST SP 800-152 2.10, 4.8, 4.9.1, 5, 6.4, 6.5, 6.6.1, 6.6.2, 6.7.1, 8.2.3, 10.1</p> <p>NIST SP 800-177 4.5, 4.6.5, 4.7, 5.1</p> <p>CCS CSC 16</p> <p>COBIT 5 DSS05.04, DSS06.03</p> <p>ISA 62443-2-1:2009 4.3.3.5.1</p> <p>ISA 62443-3-3:2013 SR 1.1, SR 1.2, SR 1.3, SR 1.4, SR 1.5, SR 1.7, SR 1.8, SR 1.9</p> <p>ISO/IEC 27001:2013 A.9.2.1, A.9.2.2, A.9.2.4, A.9.3.1, A.9.4.2, A.9.4.3</p>

Table C.1 PROTECT (PR)

Category	Subcategory	Informative References
	PR.AC-3: Remote access is managed	FIPS 140-2 Sec. 4 NIST SP 800-45 Ver. 2 9.5 NIST SP 800-53 Rev. 4 AC 17, AC-19, AC-20 NIST SP 800-57 Part 1 Rev. 4 5.3.1, 6.2.2 NIST SP 800-81-2 7.2, 9.8, 11.7.5 NIST SP 800-152 6.7.1, 8.2, 8.3 NIST SP 800-177 4.4.2.1 COBIT 5 APO13.01, DSS01.04, DSS05.03 ISA 62443-2-1:2009 4.3.3.6.6 ISA 62443-3-3:2013 SR 1.13, SR 2.6 ISO/IEC 27001:2013 A.6.2.2, A.13.1.1, A.13.2.1
	PR.AC-5: Network integrity is protected, incorporating network segregation where appropriate	OMB M-08-23 NIST SP 800-45 Ver. 2 Rev. 4 8.1.4, 9.5 NIST SP 800-53 Rev. 4 AC-4, SC-7 NIST SP 800-81-2 7.2.8, 7.9, 10.4 NIST SP 800-130 6.8.6 NIST SP 800-152 6.8.6, 8.3 NIST SP 800-177 3, 7 ISA 62443-2-1:2009 4.3.3.4 ISA 62443-3-3:2013 SR 3.1, SR 3.8 ISO/IEC 27001:2013 A.13.1.1, A.13.1.3, A.13.2.1

Table C.1 PROTECT (PR)

Category	Subcategory	Informative References
Data Security (PR.DS): Information and records (data) are managed consistent with the organization's risk strategy to protect the confidentiality, integrity, and availability of information.	PR.DS-1: Data-at-rest is protected	<p>FIPS 140-2 Sec. 4</p> <p>NIST SP 800-53 Rev. 4 SC-28</p> <p>NIST SP 800-57 Part 1 Rev. 4 4.2.5, 5.1.1, 5.2.1, 5.3.4, 5.3.5, 5.3.6, 6.2.2.3</p> <p>NIST SP 800-57 Part 2 2.2, 2.4, 3.2, 4.3, 5.3.3, 5.3.4, A.1.2, A.2.1, A.3.2</p> <p>NIST SP 800-130 1, 2.1, 2.2, 2.9, 6.1, 6.2, 6.5</p> <p>NIST SP 800-152 2.2, 4.3, 4.6, 4.7, 6.1.3, 6.4.14, 6.4.29</p> <p>CCS CSC 17</p> <p>COBIT 5 APO01.06, BAI02.01, BAI06.01, DSS06.06</p> <p>ISA 62443-3-3:2013 SR 3.4, SR 4.1</p> <p>ISO/IEC 27001:2013 A.8.2.3</p>
	PR.DS-2: Data-in-transit is protected	<p>FIPS 140-2 Sec. 4</p> <p>NIST SP 800-45 Ver. 2 All</p> <p>NIST SP 800-49 2</p> <p>NIST SP 800-52 Rev. 1 3, 4, D1.4</p> <p>NIST SP 800-53 Rev. 4 SC-8</p> <p>NIST SP 800-57 Part 1 Rev. 4 4.2.5, 5.1.1, 5.2.1, 5.3.4, 5.3.5, 5.3.6, 6.2.1.3</p> <p>NIST SP 800-57 Part 2 2.2, 5.3.3, A.2, A.3.1, A.3.2</p> <p>NIST SP 800-81-2 All</p> <p>NIST SP 800-130 1, 2.1, 2.2, 2.9, 6.1, 6.2, 6.4, 6.7.2</p> <p>NIST SP 800-152 6.1.2, 6.2.1</p> <p>NIST SP 800-177 All</p> <p>CCS CSC 17</p> <p>COBIT 5 APO01.06, DSS06.06</p> <p>ISA 62443-3-3:2013 SR 3.1, SR 3.8, SR 4.1, SR 4.2</p> <p>ISO/IEC 27001:2013 A.8.2.3, A.13.1.1, A.13.2.1, A.13.2.3, A.14.1.2, A.14.1.3</p>

Table C.1 PROTECT (PR)

Category	Subcategory	Informative References
	PR.DS-6: Integrity checking mechanisms are used to verify software, firmware, and information integrity	FIPS 140-2 Sec. 4 NIST SP 800-45 Ver. 2 2.4.2, 3, 4.2.3, 4.3, 5.1, 6.1, 7.2.2, 8.2, 9.2 NIST SP 800-49 2.2.1, 2.3.2, 3.4 NIST SP 800-52 Rev. 1 3, 4, D1.4 NIST SP 800-53 Rev. 4 SI-7 NIST SP 800-57 Part 1 Rev. 4 5.5, 6.1, 8.1.5.1, B.3.2, B.5 NIST SP 800-57 Part 2 1, 3.1.2.1.2, 4.1, 4.2, 4.3, A.2.2, A.3.2, C.2.2 NIST SP 800-81-2 All NIST SP 800-130 2.2, 4.3, 6.2.1, 63, 6.4, 6.5, 6.6.1 NIST SP 800-152 6.1.3, 6.2.1, 8.2.1, 8.2.4, 9.4 NIST SP 800-177 2.2, 4.1, 4.4, 4.5, 4.7, 5.2, 5.3 ISA 62443-3-3:2013 SR 3.1, SR 3.3, SR 3.4, SR 3.8 ISO/IEC 27001:2013 A.12.2.1, A.12.5.1, A.14.1.2, A.14.1.3
Protective Technology (PR.PT): Technical security solutions are managed to ensure the security and resilience of systems and assets, consistent with related policies, procedures, and agreements.	PR.PT-4: Communications and control networks are protected	OMB M-08-23 FIPS 140-2 Sec. 4 NIST SP 800-49 2.4.3, 2.4.4 NIST SP 800-52 Rev. 1 3, 4 NIST SP 800-53 Rev. 4 AC-4, AC-17, AC-18, CP-8, SC-7 NIST SP 800-57 Part 1 Rev. 4 5.3.1, 6.2.2 NIST SP 800-130 8.3 NIST SP 800-152 4.7, 4.11.1, 6.8.6, 8.3 CCS CSC 7 COBIT 5 DSS05.02, APO13.01 ISA 62443-3-3:2013 SR 3.1, SR 3.5, SR 3.8, SR 4.1, SR 4.3, SR 5.1, SR 5.2, SR 5.3, SR 7.1, SR 7.6 ISO/IEC 27001:2013 A.13.1.1, A.13.2.1

¹⁰ Table C.2 DETECT (DE)

Category	Subcategory	Informative References
Security Continuous Monitoring (DE.CM): The information system and assets are monitored at discrete intervals to identify cybersecurity events and verify the effectiveness of protective measures.	DE.CM-1: The network is monitored to detect potential cybersecurity events	<p>FIPS 140-2 Sec. 4</p> <p>SP 800-37 Rev. 1 3.6</p> <p>NIST SP 800-45 Ver. 2 4.1, 5.1.1, 5.1.5, 6.2.1, 6.2.2, 7.2.2</p> <p>NIST SP 800-53 Rev. 4 AC-2, AU-12, CA-7, CM-3, SC-5, SC-7, SI-4</p> <p>NIST SP 800-81-2 2, 9, 12, 13</p> <p>NIST SP 800-130 5, 6.8.5, 8.2.4, 9.8.4</p> <p>NIST SP 800-152 6.8.5, 8.2.3, 8.2.4, 8.3, 8.5</p> <p>NIST SP 800-177 3.1.1</p> <p>CCS CSC 14, 16</p> <p>COBIT 5 DSS05.07</p> <p>ISA 62443-3-3:2013 SR 6.2</p>
	DE.CM-6: External service provider activity is monitored to detect potential cybersecurity events	<p>NIST SP 800-53 Rev. 4 CA-7, PS-7, SA-4, SA-9, SI-4</p> <p>NIST SP 800-81-2 2, 9, 12, 13</p> <p>NIST SP 800-130 6.8.5, 8.2.4, 9.8.4, 12</p> <p>NIST SP 800-152 6.8.5, 8.2.3, 8.2.4, 8.3, 8.5</p> <p>ISO/IEC 27001:2013 A.14.2.7, A.15.2.1</p>
Detection Process (DE.DP): Detection processes and procedures are maintained and tested to ensure timely and adequate awareness of anomalous events.	DE.DP-4: Event detection information is communicated to appropriate parties	<p>NIST SP 800-45 Ver. 2 9.3</p> <p>NIST SP 800-53 Rev. 4 AU-6, CA-2, CA-7, RA-5, SI-4</p> <p>NIST SP 800-177 4.6</p> <p>COBIT 5 APO12.06</p> <p>ISA 62443-2-1:2009 4.3.4.5.9</p> <p>ISA 62443-3-3:2013 SR 6.1</p> <p>ISO/IEC 27001:2013 A.16.1.2</p>

¹¹ Table C.3 **RESPOND (RS)**

Category	Subcategory	Informative References
Response Planning (RS.RP): Response processes and procedures are executed and maintained, to ensure timely response to detected cybersecurity events.	RS.RP-1: Response plan is executed during or after an event	NIST SP 800-45 Ver. 2 9.3 NIST SP 800-53 Rev. 4 CP-2, CP-10, IR-4, IR-8 NIST SP 800-57 Part 1 Rev. 4 NIST SP 800-57 Part 2 3.1.2.1.3, 3.2.2.6 NIST SP 800-130 6.2.1, 6.4.5, 6.4.6, 6.8, 10.1 NIST SP 800-152 6.8, 10 NIST SP 800-177 4.6 COBIT 5 BAI01.10 CCS CSC 18 ISA 62443-2-1:2009 4.3.4.5.1 ISO/IEC 27001:2013 A.16.1.5
Communications (RS.CO): Response activities are coordinated with internal and external stakeholders, as appropriate, to include external support from law enforcement agencies.	RS.CO-2: Events are reported consistent with established criteria	NIST SP 800-45 Ver. 2 9.3 NIST SP 800-53 Rev. 4 AU-6, IR-6, IR-8 NIST SP 800-57 Part 1 Rev. 4 8.3.5, 9.3.4, 10.2.9 NIST SP 800-57 Part 2 3.1.2.1.2, 3.2.2.10, 3.2.2.14, 3.2.2.15, A.1.1, A.1.4, C.2.2.12 NIST SP 800-130 6.8 NIST SP 800-152 6.8 NIST SP 800-177 4.6 ISA 62443-2-1:2009 4.3.4.5.5 ISO/IEC 27001:2013 A.6.1.3, A.16.1.2

Table C.3 **RESPOND (RS)**

Category	Subcategory	Informative References
Mitigation (RS.MI): Activities are performed to prevent expansion of an event, mitigate its effects, and eradicate the incident.	RS.MI-1: Incidents are contained	NIST SP 800-53 Rev. 4 IR-4 NIST SP 800-130 6.8.1 NIST SP 800-152 6.8 ISA 62443-2-1:2009 4.3.4.5.6 ISA 62443-3-3:2013 SR 5.1, SR 5.2, SR 5.4 ISO/IEC 27001:2013 A.16.1.5
	RS.MI-2: Incidents are mitigated	NIST SP 800-53 Rev. 4 IR-4 NIST SP 800-57 Part 1 Rev. 4 5.3, 5.4, 5.5, 8.3.4, 8.3.5 NIST SP 800-57 Part 2 5.3.7, 5.3.8 NIST SP 800-130 4.9.3, 6.8, 9.5, 12 NIST SP 800-152 3.4.2, 4.5, 6.8, 9.5, 9.8, 12 ISA 62443-2-1:2009 4.3.4.5.6, 4.3.4.5.10 ISO/IEC 27001:2013 A.12.2.1, A.16.1.5

DRAFT

NIST CYBERSECURITY PRACTICE GUIDE

DOMAIN NAME SYSTEMS-BASED ELECTRONIC MAIL SECURITY

How-To Guides

For Security Engineers

Scott Rose

William Barker

Santos Jha

Chinedum Irrechukwu

Karen Waltermire

NIST SPECIAL PUBLICATION 1800-6C



DOMAIN NAME SYSTEMS-BASED ELECTRONIC MAIL SECURITY

1800-6C
How-To Guides
For Security Engineers

Scott Rose
Information Technology Laboratory
National Institute of Standards and Technology

William C. Barker
Dakota Consulting
Silver Spring, MD

Santos Jha
Chinedum Irrechukwu
The MITRE Corporation
McLean, VA

Karen Waltermire
National Cybersecurity Center of Excellence
National Institute of Standards and Technology

November 2016



U.S. Department of Commerce
Penny Pritzker, Secretary

National Institute of Standards and Technology
Willie May, Under Secretary of Commerce for Standards and Technology and Director

DISCLAIMER

Certain commercial entities, equipment, products, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by NIST or NCCoE, nor is it intended to imply that the entities, equipment, products, or materials are necessarily the best available for the purpose.

National Institute of Standards and Technology Special Publication 1800-6C
Natl Inst. Stand. Technol. Spec. Publ. 1800-6C, 144 pages (November 2016)
CODEN: NSPUE2

Organizations are encouraged to review all draft publications during public comment periods and provide feedback. All publications from NIST's National Cybersecurity Center of Excellence are available at <http://nccoe.nist.gov>.

Comments on this publication may be submitted to: dns-email-nccoe@nist.gov

Public comment period: November 2, 2016 through December 19, 2016

National Cybersecurity Center of Excellence
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, MD 20899
Mailstop 2002
Email: dns-email-nccoe@nist.gov

NATIONAL CYBERSECURITY CENTER OF EXCELLENCE

The National Cybersecurity Center of Excellence (NCCoE) at the National Institute of Standards and Technology (NIST) addresses businesses' most pressing cybersecurity problems with practical, standards-based solutions using commercially available technologies. The NCCoE collaborates with industry, academic, and government experts to build modular, open, end-to-end reference designs that are broadly applicable and repeatable. The center's work results in publicly available NIST Cybersecurity Practice Guides, Special Publication Series 1800, that provide users with the materials lists, configuration files, and other information they need to adopt a similar approach.

To learn more about the NCCoE, visit <http://nccoe.nist.gov>. To learn more about NIST, visit <http://www.nist.gov>.

NIST CYBERSECURITY PRACTICE GUIDES

NIST Cybersecurity Practice Guides (Special Publication Series 1800) target specific cybersecurity challenges in the public and private sectors. They are practical, user-friendly guides that facilitate the adoption of standards-based approaches to cybersecurity. They show members of the information security community how to implement example solutions that help them align more easily with relevant standards and best practices.

The documents in this series describe example implementations of cybersecurity practices that businesses and other organizations may voluntarily adopt. The documents in this series do not describe regulations or mandatory practices, nor do they carry statutory authority.

ABSTRACT

This document proposes a reference guide on how to architect, install, and configure a security platform for trustworthy email exchanges across organizational boundaries. The project includes reliable authentication of mail servers, digitally signing and encrypting email, and binding cryptographic key certificates to sources and servers. The example solutions and architectures presented here are based upon standards-based and commercially available products. The example solutions presented here can be used by any organization implementing Domain Name System-based electronic mail security.

KEYWORDS

electronic mail, digital signature; encryption; domain name system; data integrity; authentication, named entities, internet addresses, internet protocols, privacy

ACKNOWLEDGMENTS

We gratefully acknowledge the contributions of the following individuals and organizations for their generous contributions of expertise, time, and products.

Name	Organization
Nate Lesser	National Cybersecurity Center of Excellence
Karen Waltermire	National Cybersecurity Center of Excellence
Doug Montgomery	NIST ITL Advanced Networks Technologies Division
Janet Jones	Microsoft Corporation
Paul Fox	Microsoft Corporation
Joe Gersch	Secure64
Saksham Manchanda	Secure64
Benno Overeinder	NLnet Labs
Ralph Dolmans	NLnet Labs
Willem Toorop	NLnet Labs
Bud Bruegger	Fraunhofer IAO
Victoria Risk	Internet Systems Consortium
Eddy Winstead	Internet Systems Consortium

Contents

1	Introduction	1
1.1	Practice Guide Structure	2
1.2	Build Overview	3
1.3	Typographical Conventions	7
2	How to Install and Configure DNS-Protected Email Security Components	8
2.1	Laboratory Set-up	9
2.2	How to Install and Configure Microsoft Server-Based DNS-Protected Email Security Components	18
2.3	How to Install and Configure BIND	18
2.4	NSD 4 Requirements, Installation, Setup, and Configuration Components	24
2.5	How to Install and Configure OpenDNSSEC	29
2.6	Unbound	34
2.7	How to Install and Configure a DNS Signer Platform	37
2.8	How to Install and Configure a DNS Authority Platform	37
2.9	How to Install and Configure DNS Cache	37
2.10	How to Install and Configure a Dovecot/Postfix Mail Transfer Agent	38
2.11	How to Install and Configure a Thunderbird Mail Client	50
3	Device Configuration and Operating Recommendations	53
3.1	Using SSL for Cryptographic Certificate Generation	54
3.2	Cryptographic Operations (User Actions)	60
3.3	Server-to-Server Encryption Activation and Use	67
3.4	Utilities and Useful Tools	67
Appendix A	Acronyms	70
Appendix B	References	72
Appendix C	Platform Operation and Observations	75
C.1	Operations Scenarios	75
C.2	Test Sequences	76
Appendix D	Secure Name System (DNS) Deployment Checklist	89
Appendix E	Overview of Products Contributed by Collaborators	94
E.1	Open Source MUA and MTA Components	94
E.2	Microsoft Windows-Based Components	96
E.3	NLnet Labs Name Server Daemon-Based Components	98

E.4	ISC BIND Component	100
E.5	Secure64 Component	103

Appendix F Installation and Configuration Log for NSD4, Unbound, and OpenDNSSEC .. 106

Appendix G Microsoft Installation for the NCCoE 115

G.1	Microsoft Server	115
G.2	Active Directory Domain Services	116
G.3	Active Directory Certificate Services: Microsoft Certificate Authority	118
G.4	Microsoft Domain Name Services: DNS Domain Server	126
G.5	Microsoft Exchange	127

Appendix H Installation and Configuration of DNS Authority, DNS Cache, and DNS Signer at the NCCoE 144

H.1	DNS Signer	144
H.2	DNS Authority	144
H.3	DNS Cache	144

List of Figures

Figure 1.1	DNS-Based Email Security Deployment Diagram	6
Figure 2.1	DNS-Based Email Security Test Set-up	10
Figure 2.2	S/MIME and SMIMEA Deployment Flowchart	15
Figure 2.3	TLS/TLSA Deployment Flowchart.....	16
Figure 2.4	Adding Network Users for Trustworthy Email.....	17
Figure 2.5	Removing Network Users for Trustworthy Email.....	17
Figure 3.1	Example OpenSSL Configuration File.....	57
Figure C.1	Fraudulent DNS Address Spoofing Configurations.....	82
Figure C.2	Man-in-the-Middle Event Configurations	84
Figure C.3	Failed Delivery Logs	87

List of Tables

Table 2.1	Test Sequence 1	11
Table 2.2	Test Sequence 2	12
Table 2.3	Test Sequence 3	13
Table 2.4	Test Sequence 4	13
Table 2.5	Postfix Default Settings and Optional Features	44
Table C.1	Transaction Results Based on Sender TLS/DANE Connection.....	88

1 Introduction

2	1.1 Practice Guide Structure	2
3	1.2 Build Overview	3
4	1.3 Typographical Conventions	7
5		

The following guide shows IT professionals and security engineers how we implemented example solutions to the challenge of employing Domain Name System Security Extensions (DNSSEC)¹, and protocol-based digital signature and encryption technologies to protect electronic mail (email). We cover all the products that we employed in our solution set. We do not recreate the product manufacturer's documentation, which is presumed to be widely available. Rather, this guide shows how we incorporated the products together in our environment to provide composed security platforms.

Note: This is not a comprehensive tutorial. There are many possible service and security configurations for these products that are out of scope for this reference solution set.

1.1 Practice Guide Structure

This National Institute of Standards and Technology (NIST) Cybersecurity Practice Guide addresses the challenge of providing digital signature technologies to provide authentication and integrity protection for electronic mail (email) on an end-to-end basis, and confidentiality protection for email in transit between organizations.

The NIST Special Publication 1800-6 series of documents contain:

- rationale for and descriptions of a Domain Name System-Based (DNS-Based) Electronic Mail (Email) Security platform that permits trustworthy email exchanges across organizational boundaries
- a series of How-To Guides, including instructions for installation and configuration of the necessary services, that show system administrators and security engineers how to achieve similar outcomes

The solutions and architectures presented are built upon standards-based, commercially available products. These solutions can be used by any organization deploying email services that is willing to implement certificate-based cryptographic key management and DNS Security Extensions (DNSSEC). Interoperable solutions are provided that are available from different types of sources (e.g., both commercial and open source products) and function in different operating systems environments.

This summary section describes the challenge addressed by this Volume C (How-To Guide) the solution demonstrated to address the challenge, the components provided by project collaborators that have been used to compose the security platforms, an overview of how the components are configured to permit construction of platforms that cross product lines, and typographical conventions used in the Practice Guide. [Section 2, How to Install and Configure DNS-Protected Email Security Components](#), provides mail and transport layer security composition and component-centric requirements and recommendations intended to permit using Mail User Agent (MUA)², Mail Transfer Agent (MTA)³, and DNS Services components with MUAs, MTAs, and DNS Services from different vendors and open sources. It includes system requirements, installation instructions and advice and special settings requirements associated

1. RFC 4033, *DNS Security Introduction and Requirements*

2. According to NIST Special Publication (SP) 800-177, a MUA is a software component (or web interface) that allows an end user to compose and send messages and to one or more recipients. A MUA transmits new messages to a server for further processing (either final delivery or transfer to another server). See Section 2, Definitions, at <https://tools.ietf.org/html/rfc3888>.

43 with each of the MUA, MTA, and DNS Services components. In most cases where the
 44 components are commercial products, links are simply provided to vendor sites. More detailed
 45 instructions are provided for downloading, installing, and configuring open-source products.
 46 [Section 3, Device Configuration and Operating Recommendations](#), provides some specific
 47 advice and tools to support secure and reliable integration and operation of the security
 48 platforms. Topics include certificate acquisition and management options, managing mail
 49 transfer agent operation where there are significant numbers of cases of non-delivery of
 50 messages due to invalid digital signatures, device setup recommendations, email setup
 51 recommendations, and management of exception conditions. [Appendix A](#) is a list of Acronyms.
 52 [Appendix B](#) provides references. [Appendix C](#) describes test events and results from exercising
 53 different combinations of components into composed security platforms, including system
 54 responses to attempts to subvert DNSSEC protection mechanisms. [Appendix D](#) is a checklist for
 55 recommended secure domain name system deployment practices. Finally, for readers
 56 unfamiliar with any of the specific components employed by this project, [Appendix E](#) provides a
 57 set of high-level collaborator product descriptions for contributed components. [Appendix F](#)
 58 describes an example NCCoE installation and configuration of components provided by our
 59 NLnet Labs collaborator. [Appendix G](#) describes an example NCCoE installation and configuration
 60 of components provided by our Microsoft collaborator. [Appendix H](#) describes NCCoE
 61 installation and configuration of components provided by our Secure64 collaborator.

62 1.2 Build Overview

63 1.2.1 Usage Scenarios Supported

64 The scenarios supported include:

- 65 ■ “ordinary” email where the email exchanges between two organizations’ email servers
 66 communicate over Transport Layer Security (TLS)¹ with a STARTTLS² extension, and relevant
 67 TLSA³ records are published in the receiver’s DNS zone protected by DNSSEC (Scenario 1 in
 68 this document)
- 69 ■ end-to-end signed email, where the email exchanges between users in different
 70 organizations are carried over a channel protected by TLS (using the STARTTLS extension),
 71 and relevant artifacts used for signing and channel protection are published in a DNS zone
 72 protected by DNSSEC (Scenario 2). Subsequently, these artifacts are used for
 73 Secure/Multipurpose Internet Mail Extensions (S/MIME)⁴ and TLS validation.

3. Also according to SP 800-177, mail is transmitted, in a “store and forward” fashion, across networks via Mail Transfer Agents (MTAs). MTAs communicate using the Simple Mail Transfer Protocol (SMTP) described below and act as both client and server, depending on the situation. See [Section 2, Definitions](#), at <https://tools.ietf.org/html/rfc3888>.

1. RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2*

2. See RFC 3207, *SMTP Service Extension for Secure SMTP over Transport Layer Security*.

3. RFC 6698, *The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA*, Proposed Standard (August 2012; Errata) Updated by RFC 7671, RFC 7218

4. RFC 2633, *S/MIME Version 3 Message Specification*

74 In both scenarios, end-entity and personal certificates were generated from Certificate
 75 Authorities (CAs)¹. Use of “well known” (i.e. installed as trust anchors in hosts), local enterprise
 76 CAs and self-signed certificates were demonstrated.

77 While the second scenario demonstrated signing of emails, it does not include an end-to-end
 78 encrypted email scenario. Signing addresses the main security concerns in enterprise
 79 environments, which are the target of the project, but may neglect concerns of individual users
 80 who may also want to reduce information disclosure to their email providers. The two
 81 scenarios that are included may, however, serve as enablers for end-to-end encryption.
 82 Participation by parties having a primarily end-to-end encryption focus may succeed in
 83 generating industry support for the building blocks needed to support end-to-end encryption.

84 In more detail, the project's security platforms use the STARTTLS extension to include
 85 encryption of communications between two MTAs, as well as the signature of individual
 86 messages using S/MIME. The encryption and decryption with S/MIME on the end user's client
 87 was excluded from the current platform demonstration.

88 1.2.2 Architectural Overview

89 The laboratory architecture for the DNSSEC-Based Email Security project was designed to
 90 permit interconnection of Microsoft Outlook, Apple Mail, and Thunderbird MUAs with
 91 Microsoft Exchange and Postfix/Dovecot MTAs. It demonstrates the interconnection of either
 92 MTA with various DNS services contributed by collaborators. Two instantiations of each MTA
 93 type were established to demonstrate email exchanges between MTAs of the same type or
 94 different types. The various component combinations were demonstrated with three different
 95 TLSA RR² parameters: a self-signed certificate, use of local certificate authorities, and use of
 96 well-known certificate authorities.

97 [Figure 1.1](#) is a deployment diagram of the architecture used for demonstrating DNS-Based
 98 Email Security.

99 The following subsections describe the architecture's MUA, MTA, and DNS service components
 100 and Cybersecurity Framework Core categories supported by those components. Component
 101 descriptions are provided in [Appendix E](#) for those not familiar with some of the individual
 102 components.

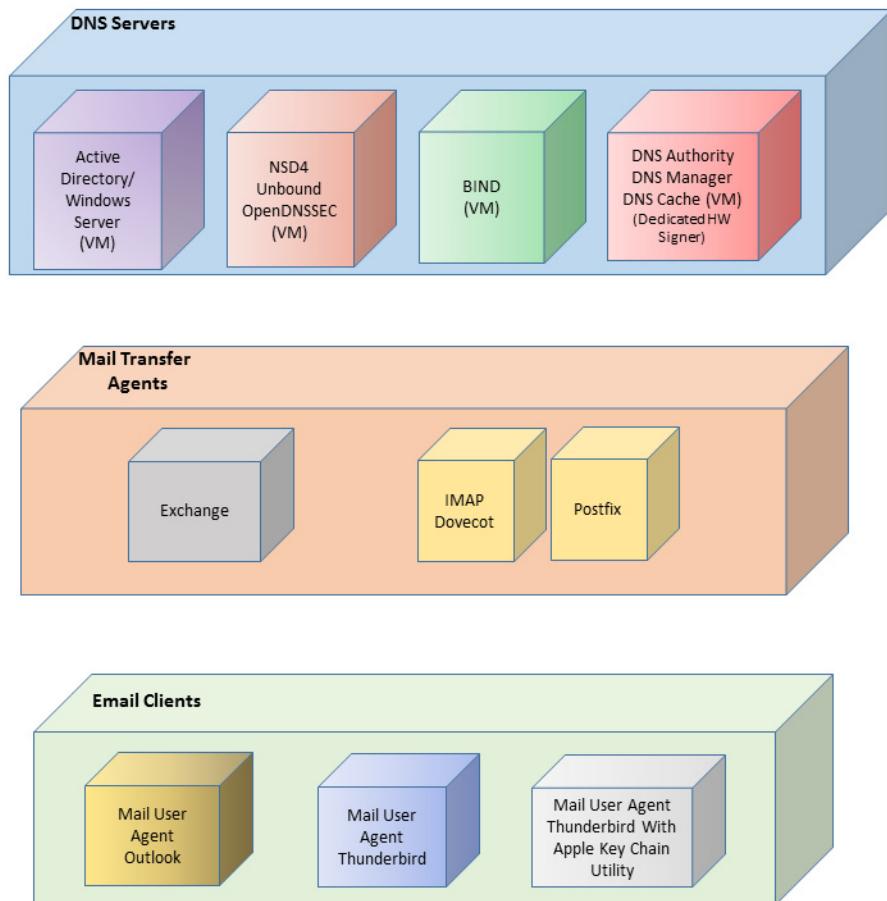
1. According to NIST SP 800-177, a trusted Certificate Authority (CA) is licensed to validate applicants' credentials, store their public key in a X.509 [RFC5280] structure, and digitally sign it with the CA's private key. TLS relies on public key cryptography and uses X.509 certificates [RFC5280] to encapsulate the public key, and the CA system to issue certificates and authenticate the origin of the key. An organization can generate its own root certificate and give its members a certificate generated from that root, or purchase certificates for each member from a well-known CA.
2. According to RFC 6698, The TLSA DNS resource record (RR) is used to associate a TLS server certificate or public key with the domain name where the record is found, thus forming a “TLSA certificate association”.

103 **1.2.2.1 Client Systems and Mail User Agents (MUAs)**

104 Client systems environments demonstrated were Microsoft Office, an open-source Linux-based
 105 Thunderbird application, and Thunderbird with a Secure64-provided Apple Key Chain utility.
 106 This set includes both commercial products and open-source software. MUA capabilities
 107 associated with the client systems are used to invoke S/MIME digital signature and signature
 108 verification for email, but user-to-user encryption is not demonstrated. Collaborators assisted
 109 in installation, integration tailoring as necessary, and testing of laboratory configurations.

110 **1.2.2.2 Email Servers**

111 Email servers include both Windows and Linux-based (Postfix/Dovecot) Mail Transfer Agents.
 112 Server-to-server encryption was demonstrated in Postfix environments. Authentication of
 113 domain and server identity was based on DNSSEC-signed DANE records. Use of these DANE
 114 records is only supported by Postfix at the time of this project. The servers were demonstrated
 115 in different DNS environments and different TLSA RR usage scenarios. In order to demonstrate
 116 representative TLSA parameters, the demonstrations used self-signed certificates, end-entity
 117 certificates generated by well-known CAs and end-entities generated by enterprise local CAs.

118 **Figure 1.1 DNS-Based Email Security Deployment Diagram**

120

1.2.2.3 DNS Servers

121 Both Windows and Linux-based DNS server and support components were contributed. DNS
 122 services provided include DNSSEC validating DNS resolvers (stub and recursive) and
 123 authoritative DNS servers for DNSSEC signed zones.¹ Support for SMIMEA and TLSA records was
 124 demonstrated. DNS components included Microsoft's Active Directory and DNS Server; Internet
 125 Systems Consortium's (ISC's) Berkeley Internet Name Domain (BIND); NLnet Labs' NSD4,
 126 Unbound, and OpenDNSSEC; and Secure64's DNS Signer, DNS Authority, DNS Cache, DNS
 127 Manager, and Apple Key Chain Utility.

128

1.3 Typographical Conventions

129 The following table presents typographic conventions used in this volume.

Typeface/ Symbol	Meaning	Example
<i>Italics</i>	filenames and pathnames references to documents that are not hyperlinks, new terms, and placeholders	For detailed definitions of terms, see the <i>NCCoE Glossary</i> .
Bold	names of menus, options, command buttons and fields	Choose File > Edit .
Monospace	command-line input, on-screen computer output, sample code examples, status codes	<code>mkdir</code>
Monospace Bold	command-line user input contrasted with computer output	service sshd start
blue text	link to other parts of the document, a web URL, or an email address	All publications from NIST's National Cybersecurity Center of Excellence are available at http://nccoe.nist.gov

1. <https://www.ietf.org/rfc/rfc1034.txt>

2 How to Install and Configure DNS-Protected Email Security Components

3	2.1	Laboratory Set-up	9
4	2.2	How to Install and Configure Microsoft Server-Based DNS-Protected Email Security Components	18
5	2.3	How to Install and Configure BIND	18
6	2.4	NSD 4 Requirements, Installation, Setup, and Configuration Components	24
7	2.5	How to Install and Configure OpenDNSSEC	29
8	2.6	Unbound.....	34
9	2.7	How to Install and Configure a DNS Signer Platform.....	37
10	2.8	How to Install and Configure a DNS Authority Platform	37
11	2.9	How to Install and Configure DNS Cache	37
12	2.10	How to Install and Configure a Dovecot/Postfix Mail Transfer Agent	38
13	2.11	How to Install and Configure a Thunderbird Mail Client.....	50
14			
15			

16 This section explains set up for the component sets provided by project collaborators. Set-up is
17 described for a virtual machine environment. The environment used for this project was the
18 Centos 7 Linux distribution running on VMware. This section includes a description of the
19 laboratory set-up for the capability demonstrations and flow charts for installation and
20 configuration of mail security and DNS security components in an enterprise. This configuration
21 overview is followed by some general instructions for installation and configuration of open
22 source components are provided, with links to source sites for more detailed instructions. Less
23 general installation is provided for commercial components, but links are provided to the
24 vendor sites. Specific installation and configuration instructions for the NCCoE environment are
25 provided as appendices ([Appendix F](#), [Appendix G](#), and [Appendix H](#)).

26 **2.1 Laboratory Set-up**

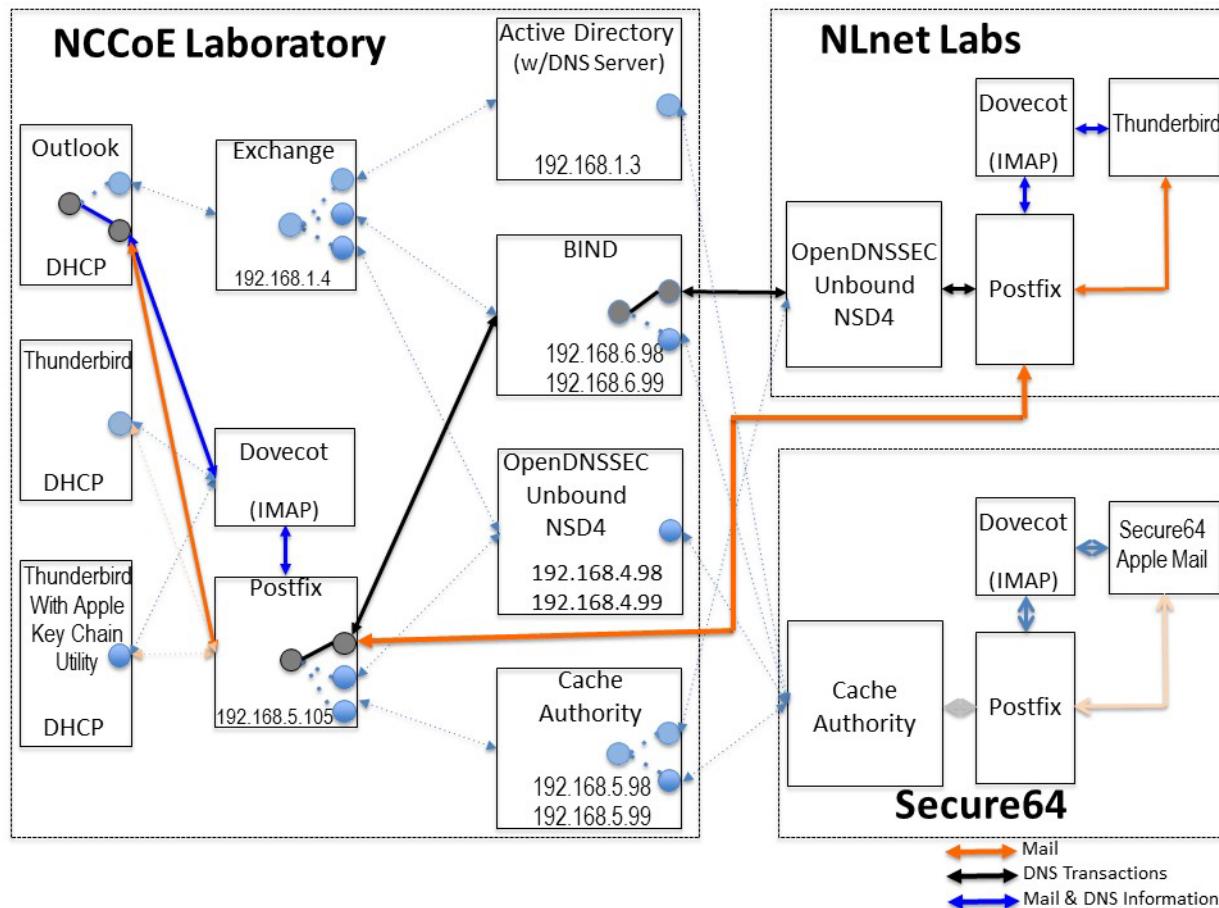
27 The design of the environment permits interconnection of components provided by different
28 collaborators (see [figure 2.1](#)).

29 The depiction shows that the project security platform test/demonstration activity was based
30 on three different clients, two MTAs, and four DNS service configurations in the lab at the
31 NCCoE exchanging messages with NLnet Labs and Secure64. All messages were signed (a mail
32 client function). Messages sent via a Postfix MTA were encrypted (server to server). The
33 message exchanges, including DNS activity will be logged at each end (lab and remote
34 correspondent).

35 The solid connectors in the depiction illustrate one case. The dotted lines depict the other cases
36 we'll want to demonstrate. A switch convention is used to reflect configuration options, but the
37 project team actually configures each component for each option.

38 The orange arrows between the mail clients and the Postfix MTA reflect the fact that clients
39 submitted email directly to the SMTP server for relay, while using Dovecot only to get mail. (The
40 depiction in [figure 2.1](#) reflects that IMAP isn't used to submit mail, only retrieve it, so the MUA
41 sent mail directly to the Postfix server, but received the reply through the Dovecot server.)

Figure 2.1 DNS-Based Email Security Test Set-up



The project team demonstrated 30 different events using various combinations of MUA, MTA, and DNS Server components divided among five test sequences. In each sequence, signed and encrypted messages were sent from a sender to a recipient. Postfix encrypted mail by default. Most of the exchanges employed either self-signed certificates or local CAs (see [Appendix C](#)). The BIND configuration was set up to obtain and validate certificates from the NIST Advanced Networks Technology Division's (ANTD's) DNS source (acting as a root CA).

49 2.1.1 Sequence 1 Set-up

50 Sequence 1 demonstrated use of well-known CA issued cryptographic certificates (CU=1),
 51 enterprise CA issued certificates (CU=2), and self-signed certificates (CU=3) with an
 52 Outlook/Exchange/Active Directory and Outlook/Exchange/BIND MUA/MTA/DNS Server stack.¹
 53 Mail was exchanged between the NCCoE and two remote sites. The first site, Secure64 in Ft
 54 Collins, Colorado, used a Thunderbird MUA with a utility for MacBook that can fetch S/MIMEA
 55 records and put them into a key store, a Postfix MTA, and Signer/Authority/Cache DNS servers.
 56 The NLnet site used an Intel-hosted Thunderbird MUA, a Postfix/Dovecot MTA, NSD4 and
 57 Unbound for processing received messages, and OpenDNSSEC for outbound messages. All
 58 messages were S/MIME signed (Scenario 2 only).

59 **Table 2.1 Test Sequence 1**

Sequence 1	NCCoE Lab			Remote Sites	Certificate on Receiver Side
Event	MUA	MTA	DNS Service	Secure64 and NLnet Labs	
1	Outlook	Exchange	Active Directory /DNS Server	Enterprise CA issued (CU=2)	Well-known CA issued (CU=1)
2	Outlook	Exchange	Active Directory /DNS Server	Same as 1	Local CA issued (CU=2)
3	Outlook	Exchange	Active Directory /DNS Server	Same as 1	Self-Signed Cert (CU=3)
4	Outlook	Exchange	BIND	Same as 1	Well-known CA issued (CU=1)
5	Outlook	Exchange	BIND	Same as 1	Local CA issued (CU=2)
6	Outlook	Exchange	BIND	Same as 1	Self-Cert (CU=3)

60 2.1.2 Sequence 2 Set-up

61 Sequence 2 demonstrated use of an Outlook/Postfix MUA/MTA configuration with a BIND DNS
 62 Server, and a Thunderbird/Postfix MUA/MTA configuration with both BIND and DNS
 63 Signer/Authority/Cache set-ups. All three certificate usage approaches were demonstrated.
 64 Mail was exchanged between the NCCoE and both Secure64 and NLnet Labs sites. As in
 65 Sequence 1, the secure64 site used a Thunderbird MUA, a Postfix MTA, and
 66 OpenDNSSEC/Unbound/NSD4 DNS servers; and the NLnet Labs site used a Thunderbird MUA, a

1. The integrity of cryptographic certificates is generally checked by verifying a digital signature generated for the certificate by its source. Certificates may be self-signed by an entity that both generates and uses it, signed by the parent enterprise that is responsible for generating and using the certificate, or be signed by some “well-known” third party certificate source that is trusted by organizations using the certificates for cryptographic protection processes. Certificate usage is designated “CU=1” for certificates issued by well-known CAs, “CU=2” for certificates issued by enterprise CAs (also known as Local CAs), and “CU=3” for certificates that are self-signed. CU=1 is generally considered most trustworthy, and CU=3 is considered least trustworthy.

67 Postfix/Dovecot MTA, NSD4 and Unbound for DNS processing received messages, and
 68 OpenDNSSEC for outbound messages. Email messages between MTAs were encrypted and
 69 successfully decrypted via TLS; an intermediate processor verified that encryption occurred;
 70 inspection of the received message verified that decryption was successful;
 71 encryption/decryption results were noted; and all messages were S/MIME signed (Scenarios 1
 72 and 2).

73 **Table 2.2 Test Sequence 2**

Sequence 2	NCCoE Lab			Remote Sites	Certificate on Receiver Side
Event	MUA	MTA	DNS Service	Secure64 and NLnet Labs	
7	Outlook	Postfix/ Dovecot	BIND	Thunderbird, Postfix/ Dovecot, NSD4/Unbound/ Open DNSSEC Self-Signed Cert (CU=3)	Well-known CA issued (CU=1)
8	Thunderbird	Postfix/ Dovecot	BIND	Same as 7	Local CA issued (CU=2)
9	Thunderbird	Postfix/ Dovecot	BIND	Same as 7	Self-Signed Cert (CU=3)
10	Thunderbird	Postfix/ Dovecot	DNS Authority/ Cache/ Signer	Same as 7	Well-known CA issued (CU=1)
11	Thunderbird	Postfix/ Dovecot	DNS Authority/ Cache/ Signer	Same as 7	Local CA issued (CU=2)
12	Thunderbird	Postfix/ Dovecot	DNS Authority/ Cache/ Signer	Same as 7	Self-Cert (CU=3)

74 2.1.3 Sequence 3 Set-up

75 Sequence 3 used an Outlook/Exchange/Active Directory stack to pose as the remote suite used
 76 in Sequence 1 and attempt to spoof an Outlook/Exchange Active Directory stack and a
 77 Thunderbird/Postfix configuration served by each of three DNS server types
 78 (OpenDNSSEC/NSD4/Unbound, DNS Signer/Authority/Cache, and BIND). All events were
 79 conducted using well-known CA and Enterprise CA-issued certificates for the impersonated
 80 sender. The email exchange between organizations was carried over TLS, and the email
 81 message was S/MIME signed on the fraudulent users' client device.

82

Table 2.3 Test Sequence 3

Sequence 3	NCCoE Lab			Remote Sites	Certificate on Receiver Side
Event	MUA	MTA	DNS Service	Secure64 and NLnet Labs	
13	Outlook	Exchange	Active Directory	Thunderbird on MacBook, Postfix/Dovecot, DNS Authority/ Cache/Signer Local CA issued (CU=2)	Local CA (CU=1)
14	Thunderbird	Postfix/ Dovecot	NSD4/ Unbound/ OpenDNSSEC	Same as 13	Local CA issued (CU=1)
15	Thunderbird on MacBook	Postfix/ Dovecot	DNS Authority/ Cache/Signer	Same as 13	Local CA issued (CU=1)
16	Outlook	Exchange	Active Directory	Same as 13	Self-Signed Cert (CU=3)
17	Thunderbird	Postfix/ Dovecot	NSD4/Unbound/ Open DNSSEC	Same as 13	Self-Signed Cert (CU=3)
18	Thunderbird	Postfix/ Dovecot	BIND	Same as 13	Self-Cert (CU=3)

83

2.1.4 Sequence 4 Set-up

84

Attempts were made to send a TLS protected email from Exchange and Postfix MTAs (in turn) to an external Postfix MTA using DNS Authority/Cache/Signer for DNS services. The NCCoE Exchange MTA used Active Directory DNS Services, and the Postfix/Dovecot MTA uses BIND, NSD4/Unbound/OpenDNSSEC, and DNS Signer/Authority/Cache DNS services. An S/MIME signed email was sent to an external Postfix MTA. Events were conducted using Well-Known CA issued certificates, events using Enterprise CA issued certificates (TLSA/SMIMEA RR parameter of CU=2) for TLS and S/MIME on the receiver side, and three using self-signed certificates (TLSA/SMIMEA RR parameter of CU=3) for TLS and S/MIME on the receiver side. An Outlook/Exchange/Active Directory stack acted as a man-in-the-middle and attempted to impersonate the legitimate receiver.

94

Table 2.4 Test Sequence 4

Sequence 3	NCCoE Lab			Legitimate Remote Site	Certificate on Receiver Side
Event	MUA	MTA	DNS Service		
19	Outlook	Exchange	Active Directory	Secure64	Well-Known CA (CU=1)
20	Thunderbird	Exchange	BIND	Secure64	Well-Known CA (CU=1)
21	Thunderbird	Postfix	NSD4/Unbound/ Open DNSSEC	Secure64	Well-Known CA (CU=1)

Table 2.4 Test Sequence 4

Sequence 3	NCCoE Lab			Legitimate Remote Site	Certificate on Receiver Side
Event	MUA	MTA	DNS Service		
22	Thunderbird on MacBook	Postfix/ Dovecot	DNS Authority/ Cache/Signer	Secure64	Well-Known CA (CU=1)
23	Outlook	Exchange	Active Directory	Secure64	Local CA (CU=2)
24	Thunderbird	Postfix/ Dovecot	BIND	Secure64	Local CA (CU=2)
25	Thunderbird on MacBook	Postfix/ Dovecot	NSD4/Unbound/ Open DNSSEC	Secure64	Local CA (CU=2)
26	Thunderbird on MacBook	Postfix/ Dovecot	DNS Authority/ Cache/Signer	Secure64	Local CA (CU=2)
27	Thunderbird	Postfix/ Dovecot	Active Directory	Secure64	Self-Cert (CU=3)
28	Thunderbird	Exchange	BIND	Secure64	Self-Cert (CU=3)
29	Thunderbird on MacBook	Postfix/ Dovecot	NSD4/Unbound/ Open DNSSEC	Secure64	Self-Cert (CU=3)

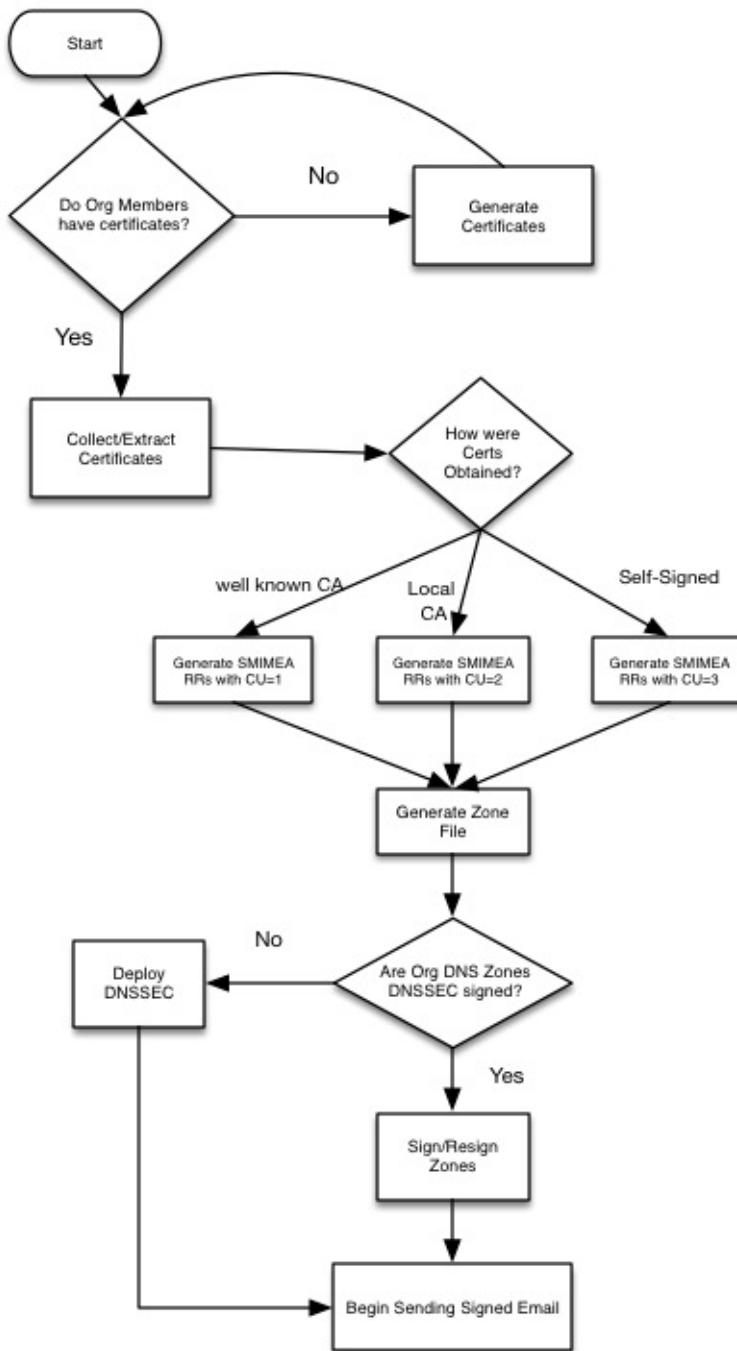
95 2.1.5 Sequence 5 Set-up

96 This sequence used an Authoritative DNS Server, a DANE-aware Postfix server, and four
 97 Exchange MTAs (each set up differently). One ran without TLSA, one had good TLSA and a
 98 self-signed certificate (CU=3), one had bad PKIX and a certificate from a well-known CA (CU=1),
 99 and one had a bad TLSA with a self-signed certificate (CU=3). A script running on the Postfix
 100 server generates a message stream. Logs of failed DNS events were examined.

101 2.1.6 How to Deploy SMIMEA and TLSA Software for Trustworthy Email

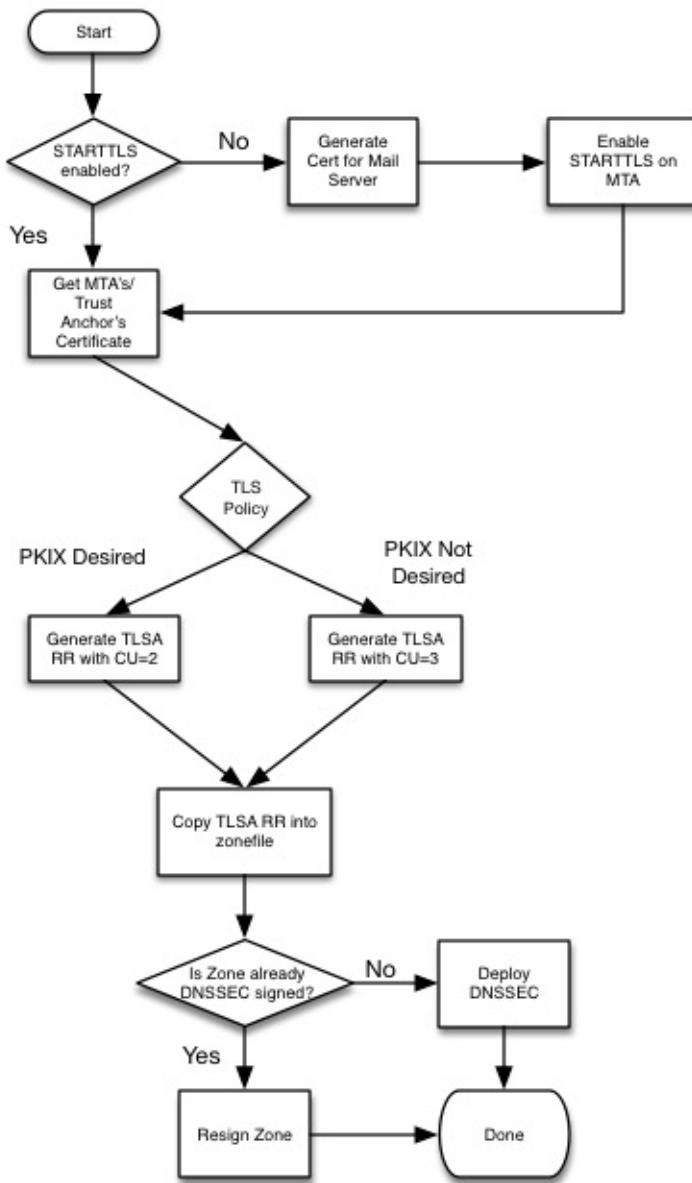
102 Set-up for the test sequences required deploying SMIMEA and TLSA, and adding certificates and
 103 records for users. Figures 3 and 4 are flowcharts depicting the steps required for installation
 104 and configuration of MUAs, MTAs, and DNS servers necessary to trustworthy email. [Figure 2.2](#)
 105 depicts the process for setting up secure/multipurpose Internet mail extensions (S/MIME and
 106 SMIMEA). [Figure 2.3](#) depicts the process for setting up transport layer security (i.e., TLS and
 107 TLSA). The figures assume that the enterprise has deployed DNSSEC, including DANE-aware
 108 components. The figures include questions regarding the installation and configuration status
 109 of components, and provides recommendations based on the answers to those questions.
 110 Together with the Secure Name System (DNS) Deployment Checklist provided as [Appendix D](#),
 111 these flowcharts are intended to facilitate establishment of a trustworthy email capability in a
 112 wide range of environments.

Figure 2.2 S/MIME and SMIMEA Deployment Flowchart



115

Figure 2.3 TLS/TLSA Deployment Flowchart



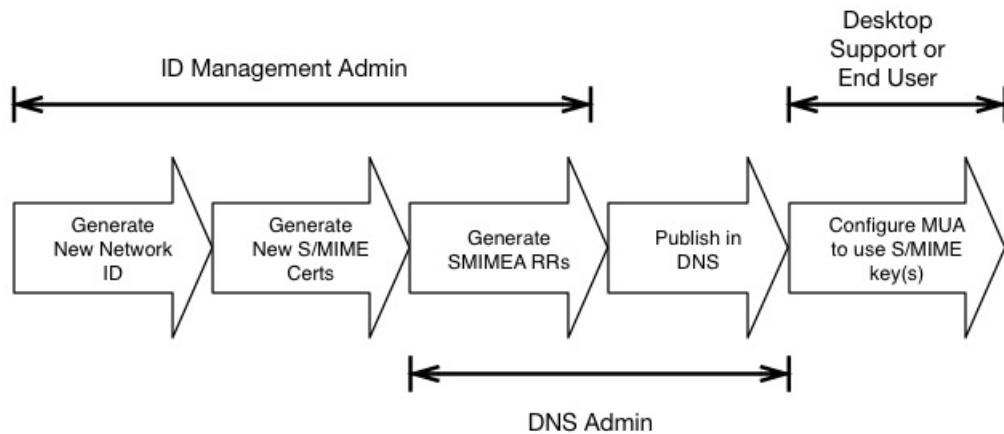
116

117 2.1.7 Adding and Removing Network Users

118
119
120
121
122

Adding users to networks with trustworthy email enabled involves identity management administrative, DNS administrative, and end user support activities. Figure 2.4 depicts the process for generating user network identities, new S/MIME Certificates for users, and SMIMEA resource records; publishing the records in the DNS, and configuring users' MUAs to use S/MIME keys.

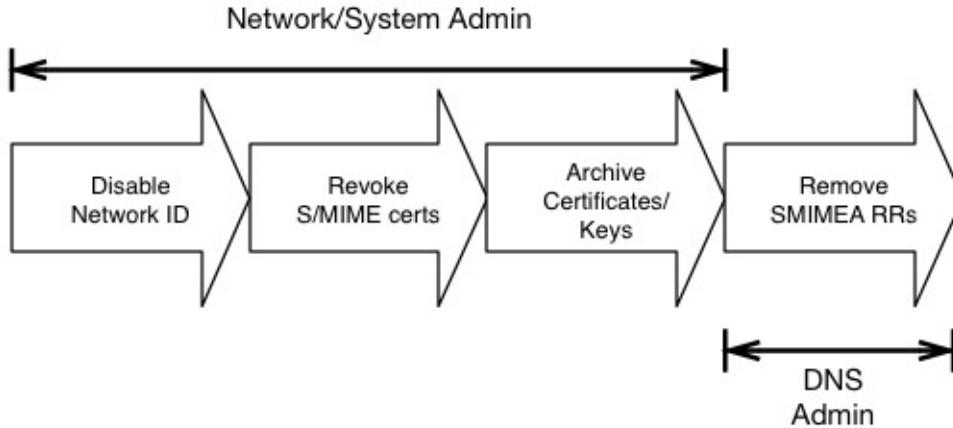
123

Figure 2.4 Adding Network Users for Trustworthy Email

124

125 When a user leaves an organization or access to network resources is revoked for other reasons,
 126 it is necessary to revoke the credentials that associate the user with the organization. This
 127 action requires the network or system administrator to disable the user's network ID, revoke
 128 the user's S/MIME certificates, and archive the certificates and associated keys; and requires
 129 the DNS administrator to remove the user's SMIMEA resource records (RRs). [Figure 2.5](#) depicts
 130 the flow for this process.

131

Figure 2.5 Removing Network Users for Trustworthy Email

132

2.2 How to Install and Configure Microsoft Server-Based DNS-Protected Email Security Components

Outlook, Exchange, Active Directory, and DNS Server are commercial products that can be accessed from Microsoft's web (e.g., <https://www.microsoft.com/en-us/>). Outlook is generally bundled in Microsoft Office (e.g., Office365 for Windows 10), and DNS Server is bundled in Microsoft Server systems (e.g., Server 2016). Active Directory tools and applications are not installed in Windows 10 by default, but instructions regarding how to get them can be found at <http://www.technipages.com/windows-install-active-directory-users-and-computers>. DNS Server is bundled with Server 2016. Please note that IP addresses, domain names, and mail addresses are, in many cases, specific to the NCCoE laboratory configuration and must not be used in actual implementations.

2.2.1 Installation Basics and System Requirements

System requirements are product-specific, and installation instructions are highly dependent of version, intended configuration, and tools set employed. The installation process, tools employed, and configuration process followed in setting up the NCCoE Microsoft components are provided as [Appendix G](#) to this Practice Guide. Manual pages are provided for individual applications of products and tools (e.g., [https://technet.microsoft.com/en-us/library/bb245702\(v=exchg.80\).aspx](https://technet.microsoft.com/en-us/library/bb245702(v=exchg.80).aspx) and [https://technet.microsoft.com/en-us/library/bb123543\(v=exchg.141\).aspx](https://technet.microsoft.com/en-us/library/bb123543(v=exchg.141).aspx) for Exchange, [https://technet.microsoft.com/en-us/library/dn626158\(v=exchg.150\).aspx](https://technet.microsoft.com/en-us/library/dn626158(v=exchg.150).aspx) for Outlook), and [https://technet.microsoft.com/en-us/library/cc732284\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc732284(v=ws.11).aspx) for configuring a DNS server for use with Active Directory domain services; and from a wide variety of third party sources.

2.2.2 Installation of Active Directory, Server, and Exchange in the NCCoE Configuration

Appendix G describes installation and configuration of Active Directory, Server, and Exchange at the NCCoE.

2.3 How to Install and Configure BIND¹

The current guide for getting started with BIND and instruction on how to build and run named with a basic recursive configuration can be found at <https://kb.isc.org/article/AA-00768/46/Getting-started-with-BIND-how-to-build-and-run-named-with-a-basic-recursive-configuration.html>. The current BIND 9 Reference Manual can be found at https://www.isc.org/wp-content/uploads/2014/01/Bv910ARM.pdf&hl=en_US. An overview of installation and configuration basics follow. Please note that IP addresses, domain names, and mail addresses are, in many cases, specific to the NCCoE laboratory configuration and must not be used in actual implementations.

169 2.3.1 Installation Basics and System Requirements

170 The NCCoE BIND installation was based on Centos 7. ISC specifies that BIND 9 currently requires
171 a UNIX system with an ANSI C compiler, basic POSIX support, and a 64 bit integer type.

172 ISC has also had success in building and testing on the following systems:

- 173 ■ COMPAQ Tru64 UNIX 5.1B
- 174 ■ Fedora Core 6
- 175 ■ FreeBSD 4.10, 5.2.1, 6.2
- 176 ■ HP-UX 11.11
- 177 ■ Mac OS X 10.5
- 178 ■ NetBSD 3.x, 4.0-beta, 5.0-beta
- 179 ■ OpenBSD 3.3 and up
- 180 ■ Solaris 8, 9, 9 (x86), 10
- 181 ■ Ubuntu 7.04, 7.10
- 182 ■ Windows XP/2003/2008

183 ISC also has recent reports from the user community that a supported version of BIND will build
184 and run on the following systems:

- 185 ■ AIX 4.3, 5L
- 186 ■ CentOS 4, 4.5, 5
- 187 ■ Darwin 9.0.0d1/ARM
- 188 ■ Debian 4, 5, 6
- 189 ■ Fedora Core 5, 7, 8
- 190 ■ FreeBSD 6, 7, 8
- 191 ■ HP-UX 11.23 PA
- 192 ■ MacOS X 10.5, 10.6, 10.7
- 193 ■ Red Hat Enterprise Linux 4, 5, 6
- 194 ■ SCO OpenServer 5.0.6
- 195 ■ Slackware 9, 10
- 196 ■ SuSE 9, 10

197 Note: As of BIND 9.5.1, 9.4.3, and 9.3.6, older versions of Windows, including Windows NT and
198 Windows 2000, are no longer supported.

1. This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (<http://www.openssl.org/>), cryptographic software written by Eric Young (eay@cryptsoft.com), and software written by Tim Hudson (tjh@cryptsoft.com).

199 Information regarding downloading BIND can be found at
200 <https://www.isc.org/downloads/bind/>

201 2.3.2 BIND Installation and Configuration

202 ISC's recommended link for BIND starter information is:
203 <https://kb.isc.org/article/AA-00768/46/Getting-started-with-BIND-how-to-build-and-run-nam>
204 [ed-with-a-basic-recursive-configuration.html](https://kb.isc.org/article/AA-00768/46/Getting-started-with-BIND-how-to-build-and-run-nam). For authoritative configuration, refer to the
205 BIND9 ARM (<https://www.isc.org/downloads/bind/doc/bind-9-10/>).

206 To build, just enter:

207 `./configure`
208 `make`

209 Do not use a parallel “make”.

210 2.3.2.1 Environmental Variables

211 Several BIND environment variables that can be set before running configure will affect
212 compilation:

213 ■ **CC**

214 The C compiler to use. configure tries to figure out the right one for supported systems.

215 ■ **CFLAGS**

216 C compiler flags. Defaults to include -g and/or -O2 as supported by the compiler. Please
217 include '-g' if you need to set **CFLAGS**.

218 ■ **STD_CINCLUDES**

219 System header file directories. Can be used to specify where add-on thread or IPv6 support
220 is, for example. **STD_CINCLUDES** defaults to empty string.

221 ■ **STD_CDEFINES**

222 Any additional preprocessor symbols you want defined. **STD_CDEFINES** defaults to empty
223 string.

224 Possible settings:

- 225 • Change the default syslog facility of **named/lwresd**.

226 `-DISC_FACILITY=LOG_LOCAL0`

- 227 • Enable DNSSEC signature chasing support in **dig**.

228 `-DDIG_SIGCHASE=1` (sets `-DDIG_SIGCHASE_TD=1` and
229 `-DDIG_SIGCHASE_BU=1`)

- 230 • Disable dropping queries from particular well known ports.

231 `-DNS_CLIENT_DROPPORT=0`

- 232 • Sibling glue checking in named-checkzone is enabled by default.

233 To disable the default check set. `-DCHECK_SIBLING=0`.

234 • named-checkzone checks out-of-zone addresses by default.

235 To disable this default set `-DCHECK_LOCAL=0`.

236 • To create the default pid files in `${localstatedir}/run` rather than `${localstatedir}/run/{named, lwresd}` set `-DNS_RUN_PID_DIR=0`

237

238 • Enable workaround for Solaris kernel bug about `/dev/poll`

239 `-DISC_SOCKET_USE_POLLWATCH=1`

240 • The watch timeout is also configurable, e.g.,

241 `-DISC_SOCKET_POLLWATCH_TIMEOUT=20`

242 ■ **LDFLAGS**

243 Linker flags. Defaults to empty string.

244 2.3.2.2 Cross Compiling

245 The following need to be set when cross compiling:

- 246 ■ **BUILD_CC**
- 247 The native C compiler.
- 248 ■ **BUILD_CFLAGS** (optional)
- 249 ■ **BUILD_CPPFLAGS** (optional)
- 250 Possible Settings:
- 251 `-DNEED_OPTARG=1` (optarg is not declared in `<unistd.h>`).
- 252 ■ **BUILD_LDFLAGS** (optional)
- 253 ■ **BUILD_LIBS** (optional)

254 2.3.2.3 Multithreading Support

255 On most platforms, BIND 9 is built with multithreading support, allowing it to take advantage of
 256 multiple CPUs. You can configure this by specifying `--enable-threads` or `--disable-threads`
 257 on the configure command line. The default is to enable threads, except on some older
 258 operating systems on which threads are known to have had problems in the past.

259 Note: *Prior to BIND 9.10, the default was to disable threads on Linux systems; this has been*
 260 *reversed. On Linux systems, the threaded build is known to change BIND's behavior with*
 261 *respect to file permissions; it may be necessary to specify a user with the -u option when running*
 262 *named.*

263 2.3.2.4 Shared Libraries

264 To build shared libraries, specify `--with-libtool` on the configure command line.

265 2.3.2.5 Large Servers

266 Certain BIND compiled-in constants and default settings can be increased to values better
267 suited to large servers with abundant memory resources (e.g., 64-bit servers with 12G or more
268 of memory) by specifying `--with-tuning=large` on the configure command line. This can
269 improve performance on big servers, but will consume more memory and may degrade
270 performance on smaller systems.

271 2.3.2.6 DNSSEC Support

272 For the BIND server to support DNSSEC, you need to build it with crypto support. You must have
273 OpenSSL 0.9.5a or newer installed and specify `--with-openssl` on the configure command
274 line. If OpenSSL is installed under a nonstandard prefix, you can tell configure where to look for
275 it using `--with-openssl=/prefix`.

276 2.3.2.7 HTTP Statistics Channel Support

277 To support the HTTP statistics channel, the BIND server must be linked with at least one of the
278 following: libxml2 (<http://xmlsoft.org>) or json-c (<https://github.com/json-c>). If these are
279 installed at a nonstandard prefix, use `--with-libxml2=/prefix` or `--with-libjson=/prefix`.

280 To support compression on the HTTP statistics channel, the server must be linked against libzlib
281 (`--with-zlib=/prefix`).

282 2.3.2.8 Python Support

283 Python requires '`argparse`' and '`ply`' to be available. '`argparse`' is a standard module as of
284 Python 2.7 and Python 3.2.

285 2.3.2.9 Files Larger than 2GB

286 On some platforms it is necessary to explicitly request large file support to handle files bigger
287 than 2GB. This can be done by `--enable-largefile` on the BIND configure command line.

288 2.3.2.10 Fixed rrset-order Option

289 Support for the `fixed` rrset-order option can be enabled or disabled by specifying
290 `--enable-fixed-rrset` or `--disable-fixed-rrset` on the BIND configure command line. The
291 default is `disabled`, to reduce memory footprint.

292 2.3.2.11 IPv6 Support

293 If your operating system has integrated support for IPv6, it will be used automatically. If you
294 have installed KAME IPv6 separately, use `--with-kame[=PATH]` to specify its location.

295 2.3.2.12 Installing named and BIND 9 Libraries

296 The `make install` tool will install `named` and the various BIND 9 libraries. By default, installation
297 is into `/usr/local`, but this can be changed with the `--prefix` option when running `configure`.

2.3.2.13 Directory Setting Options

You may specify the option `--sysconfdir` to set the directory where configuration files like `named.conf` go by default, and `--localstatedir` to set the default parent directory of `run/named.pid`. For backwards compatibility with BIND 8, `--sysconfdir` defaults to `/etc` and `--localstatedir` defaults to `/var` if no `--prefix` option is given. If there is a `-prefix` option, `sysconfdir` defaults to `$prefix/etc` and `localstatedir` defaults to `$prefix/var`.

2.3.2.14 Other Configure Options

To see additional configure options, run `configure --help`. Note that the help message does not reflect the BIND 8 compatibility defaults for `sysconfdir` and `localstatedir`. If you're planning on making changes to the BIND 9 source, you should also `make depend`. If you're using Emacs, you might find `make tags` helpful.

2.3.2.15 Re-running Configure

If you need to re-run configure please run `make distclean` first. This will ensure that all the option changes take.

2.3.2.16 Building with gcc

Building with gcc is not supported, unless gcc is the vendor's usual compiler (e.g. the various BSD systems, Linux).

2.3.2.17 Known Compiler and OS Issues

Known compiler issues include the following:

- gcc-3.2.1 and gcc-3.1.1 is known to cause problems with solaris-x86.
- gcc prior to gcc-3.2.3 ultrasparc generates incorrect code at -O2.
- gcc-3.3.5 powerpc generates incorrect code at -O2.
- Irix, MipsPRO 7.4.1m is known to cause problems.
- SunOS 4 requires `printf` to be installed to make the shared libraries.
- sh-utils-1.16 provides a `printf` which compiles on SunOS 4.
- Linux requires kernel

2.3.3 Testing

A limited BIND test suite can be run with `make test`. Many of the tests require you to configure a set of virtual IP addresses on your system, and some require Perl. (See `bin/tests/system/README` for details).

2.3.4 BIND Documentation

The [BIND 9 Administrator Reference Manual](#) is included with the source distribution in DocBook XML and HTML format, in the `doc/arm` directory.

331 Some of the programs in the BIND 9 distribution have man pages in their directories. In
332 particular, the command line options of **named** are documented in **/bin/named/named.8**.

333 There is now also a set of man pages for the **lwres** library.

334 For upgrading from BIND 8, please read the migration notes in **doc/misc/migration**. If you are
335 upgrading from BIND 4, read **doc/misc/migration-4to9**.

336 Frequently asked questions and their answers can be found in **FAQ**.

337 Additional information on various subjects can be found in the other **README** files.

338 2.3.5 BIND Support

339 Although BIND is open source software, support is available from ISC.

340 2.4 NSD 4 Requirements, Installation, Setup, and 341 Configuration Components

342 The links for NSD 4.1.13 tar files, manual pages, and SVN repository can be found at
343 <https://www.nlnetlabs.nl/projects/nsd/>. This repository provides for downloading of the latest
344 NSD 4 version. NSD 4 can be installed on Unix-based systems (e.g., FreeBSD, OpenBSD, NetBSD,
345 Mac OS X, and Solaris), including Linux systems such as Red Hat Enterprise, Centos, Debian,
346 Ubuntu, and Gentoo. Please note that IP addresses, domain names, and mail addresses are, in
347 many cases, specific to the NCCoE laboratory configuration and must not be used in actual
348 implementations.

349 2.4.1 NSD 4 Installation Basics

350 NSD4 is available in distribution repositories such that a package manager can install it with a
351 single command:

352 For Red Hat Enterprise and Centos (Centos 7 was used in the NCCoE example):

353 `yum install nsd`

354 For Debian and Ubuntu:

355 `sudo apt-get install nsd`

356 For Gentoo:

357 `emerge nsd`

358 2.4.2 NSD 4 Configuration (nsd.conf)

359 Different paths exist for NSD4 (nsd.conf). Their paths depend on your distribution:

360 Centos - Red Hat Enterprise: `/etc/nsd/nsd.conf`

361 Debian - Ubuntu: `/etc/nsd/nsd.conf`

2.4.2.1 Master Configuration

The following is a master configuration for NSD4 for a Centos system. This example shows nsd4 serving the domain dnslabs.dnsops.gov on the IP address 129.6.45.38. The log file for the actual NCCoE installation and configuration of NSD4 with Unbound and OpenDNSSEC for the DNS-Based Email Security project is provided as [Appendix F](#).

```
362 #
363 # nsd.conf -- the NSD(8) configuration file, nsd.conf(5).
364 #
365 # Copyright (c) 2001-2011, NLnet Labs. All rights reserved.
366 #
367 # See LICENSE for the license.
368 #
369 #
370 # This is a configuration file commented out, you just need to change
371 # the IP and the zone file to customize it.
372 #
373 #
374 #
375 # options for the nsd server
376 server:
377 # uncomment to specify specific interfaces to
378 bind (default wildcard interface).
379 # ip-address: localhost
380 ip-address: 129.6.45.38
381 #
382 #
383 #
384 # don't answer VERSION.BIND and VERSION.SERVER
385 # CHAOS class queries
386 # Keep yes for security reasons.
387 hide-version: yes
388 #
389 #
390 # enable debug mode, does not fork daemon process into the background.
391 # debug-mode: no
392 #
393 # do-ip4
394 default: yes
395 #
396 # do-ip6
397 default: yes
398 #
399 # Enable IPv6 as advice.
400 #
401 # the database to use, this is the standard path.
402 # disable database mode. Explicitly set database: ""
403 # database: ""
```

```
405 # identify the server (CH TXT ID.SERVER entry).
406 identity: ""
407
408 # NSID identity (hex string). default disabled.
409 # nsid: "aabbccdd"
410
411 # log messages to file. Default to stderr and
412 # syslog (with facility LOG_DAEMON).
413 # logfile: "/var/log/nsd.log"
414
415 # Number of NSD servers to fork, keep 1 for low
416 # memory VPS
417 server-count: 1
418
419 # Maximum number of concurrent TCP connections
420 # per server.
421 # This option should have a value below 1000, 10
422 # is good for a low memory VPS
423 tcp-count: 10
424
425 # Maximum number of queries served on a single
426 # TCP connection.
427 # By default 0, which means no maximum.
428 # tcp-query-count: 0
429
430 # Override the default (120 seconds) TCP timeout.
431 # tcp-timeout: 120
432
433 # Preferred EDNS buffer size for IPv4.
434 # ipv4-edns-size: 4096
435
436 # Preferred EDNS buffer size for IPv6.
437 # ipv6-edns-size: 4096
438
439 # File to store pid for nsd in.
440 # pidfile: "/var/run/nsd/nsd.pid"
441
442 # port to answer queries on. default is 53.
443 # port: 53
444
445 # statistics are produced every number of
446 # seconds.
447 # statistics: 3600
```

```

448
449     # if per zone statistics is enabled, file to
450     # store statistics.
451     # zone-stats-file: "/var/log/nsd.stats"
452
453     # The directory for zonefile: files.
454     zonesdir: "/etc/nsd/zones"
455
456     #This is the definition of the first zone, you
457     # must have 1 for every domain.
458     zone:
459         name: dnslabs.dnsops.gov
460         #file in the zonesdir that contains the domain
461         #information.
462         zonefile: dnslabs.dnsops.gov.conf
463
464     # See https://www.nlnetlabs.nl/projects/nsd/nsd-control.8.html for
465     nsd-control config

```

2.4.2.2 NSD Zone File

The next step is setting up zone files. The following instructions set up a simple zone file that just defines the SOA, the NS, MX and some address for the domain:

```

469 ;## NSD authoritative only DNS
470
471 $ORIGIN dnslabs.dnsops.gov. ; default zone domain
472 $TTL 86400 ; default time to live
473
474 @ IN SOA nev1 admin@dnslabs.dnsops.gov (
475     2012082703 ; serial number
476     28800 ; Refresh
477     14400 ; Retry
478     864000 ; Expire
479     86400 ; Min TTL
480 )
481
482 NS nev1.dnslabs.dnsops.gov .
483 NS nev2.dnslabs.dnsops.gov .
484 MX 10 mail.dnslabs.dnsops.gov .
485
486 mail IN A 129.6.45.38
487 www IN A 129.6.45.38
488 nev1 IN A 129.6.45.38
489 nev2 IN A 129.6.45.38

```

```

490      * IN A 129.6.45.38
491      @ IN A 129.6.45.38
492
493      ;## NSD authoritative only DNS
494
495      For NSD it is a requisite to set your NS name server hostname (nev1.dnslabs.dnsops.gov to
496      129.6.45.38 in this example) to the same IP address NSD is listening on, the one we have set in
497      the nsd.conf file. This is so important because a resolving DNS server, like BIND, will ask NSD
498      what the current authoritative name server IP address is. NSD will say the name server for
499      dnslabs.dnsops.gov is nev1.dnslabs.dnsops.gov and its IP is 129.6.45.38. And so
500      129.6.45.38 is the address that another service like BIND will use to connect.
501      * IN A 129.6.45.38
502
503      includes the names in the domain .dnslabs.dnsops.gov.

```

503 2.4.2.3 Compile the NSD Database and Start Daemon

504 Note: *NLnet Labs advises against running NSD4 in the database mode unless there is a*
 505 *compelling local reason.*

- 506 1. General


```

507     Nsd-control stop/start
      
```
- 508 2. Restart Command: If a message is received that there are errors in the zone file, correct
 509 them; otherwise restart as follows:
 - 510 a. For Red Hat or Centos Server:


```

511          /etc/init.d/nsd restart
          
```
 - 512 b. For Debian or Ubuntu server:


```

513          /etc/init.d/nsd4 restart
          
```

514 Note: A *restart* is not needed to *reload zonefile*. Use *reload* or *reconfig*.

515 2.4.2.4 Testing NSD4

516 The easiest way to test the NSD4 configuration is to run a **dig** from the resolver querying the
 517 NSD server for the domain you just defined, such as:

```
518 dig @129.6.45.38 dnslabs.dnsops.gov
```

519 The output should look something like the following:

```

520      ; <> ; DIG 9.3.6-20.P1.e15_8.2 ;
521      <> @129.6.45.38 dnslabs.dnsops.gov
522      ; 1(1 server found)
523      ; ; global options: printcmd
524      ; ; Got answer:
525      ; ; ->>HEADER<
      
```

526 In this output you should see in the **answer** section the correct association between your DNS
 527 name and IP, and in the **AUTHORITY** section the correct association between your NS and the
 528 configured IP.

529 **2.4.2.5 NSD4 Support**

530 Although NSD4 is open source software, support is available from NLnet Labs via its subsidiary
 531 Open Netlabs (<http://www.opennetlabs.com>).

532 **2.5 How to Install and Configure OpenDNSSEC**

533 The log file for an actual NCCoE installation and configuration of OpenDNSSEC with Unbound
 534 and NSD4 for the DNS-Based Email Security project is provided as Appendix F. For cryptographic
 535 operations, OpenDNSSEC uses the PKCS#11 interface supported by hardware security modules
 536 (HSMs). As an alternative to real HSMs, the OpenDNSSEC project developed SoftHSM, a drop-in
 537 replacement that uses the Botan or OpenSSL cryptographic library. SQLite or MySQL can be
 538 used as database back-ends. It is used on the .se, .dk, .nl, .ca, and .uk top-level domains and
 539 more. OpenDNSSEC can be downloaded from:

- 540 ■ <https://dist.opendnssec.org/source/opendnssec-2.0.1.tar.gz>
- 541 ■ <https://dist.opendnssec.org/source/opendnssec-2.0.1.tar.gz.sig>
- 542 ■ Checksum SHA256:
543 bf874bbb346699a5b539699f90a54e0c15fff0574df7a3c118abb30938b7b346

544 Please note that IP addresses, domain names, and mail addresses are, in many cases, specific to
 545 the NCCoE laboratory configuration and must not be used in actual implementations.

546 **2.5.1 OpenDNSSEC Installation Basics and System Requirements**

547 OpenDNSSEC¹ will run on most Linux, BSD and Solaris operating systems. The community
 548 provides binary packages for several platforms to assist installation. This Practice Guide,
 549 however, assumes those packages are not available. If you have found an appropriate system to
 550 run OpenDNSSEC on, it is time to install its dependencies. OpenDNSSEC relies on a database
 551 backend and currently supports MySQL and SQLite. MySQL is recommended because SQLite
 552 doesn't scale well and has some known locking issues. Furthermore, OpenDNSSEC depends on:

- 553 ■ ldns, version 1.6.12 and up with the exceptions of 1.6.14 and 1.6.15
- 554 ■ libxml2, libxml2-dev, libxml2

555 As indicated above, OpenDNSSEC generally assumes use of a cryptographic Hardware Security
 556 Module (HSM) via the PKCS#11 interface. An alternative is use of SoftHSM, a software-only
 557 implementation of an HSM. SoftHSM depends on Botan (a cryptographic library) version 1.8.5
 558 or greater, or OpenSSL (for SoftHSM 2.0 and higher), and SQLite version 3.3.9 or greater. Install
 559 SoftHSM (<https://www.opendnssec.org/2016/03/softhsm-2-1-0/>) with:

560 `$ tar -xzf softhsm-X.Y.Z.tar.gz`

1. <https://www.opendnssec.org/>.

```

561 $ cd softHSM-X.Y.Z $ ./configure
562 $ make
563 $ sudo make install

564 By default, the binary will be installed in /usr/local/bin/ and the configuration is expected
565 to be at /etc/softHSM.conf. Open the file and specify a slot for OpenDNSSEC. For example:
566 # SoftHSM slots 0:/var/lib/softHSM/slot0.db

567 The token database does not exist at this stage. It is necessary to initialize it with:
568 $ softHSM --init-token --slot 0 --label "OpenDNSSEC"

569 When prompted, fill in a SO (Security Officer) PIN and user PIN. Remember it, you will need to
570 configure it for OpenDNSSEC. The SO PIN can be used to reinitialize the token. The user PIN is
571 handed out to OpenDNSSEC. If your company does not have a SO, just pick the same PIN for
572 both roles.

573 Make sure OpenDNSSEC has permission to access the token database.
574 $ chown opendnssec /var/lib/softHSM/slot0.db
575 $ chgrp opendnssec /var/lib/softHSM/slot0.db

```

576 2.5.2 OpenDNSSEC Installation

577 While the log file for an actual installation and configuration of OpenDNSSEC with Unbound and
 578 NSD4 for the DNS-Based Email Security project is provided as Appendix F, some more general
 579 information regarding OpenDNSSEC installation¹ follows:

580 Run these commands to install OpenDNSSEC:

```

581 $ tar -xzf opendnssec-X.Y.Z.tar.gz
582 $ cd OpenDNSSEC-X.Y.Z $ ./configure
583 $ make
584 $ make install

```

585 By default, the binaries will be installed in /usr/local/bin/ and /usr/local/sbin/. The
 586 configuration files are located in the /etc/opendnssec/ directory. The working directories are
 587 under /var/opendnssec/.

588 2.5.3 OpenDNSSEC Configuration Requirements

589 The default configuration installs default values for entities that just wants to sign their domains
 590 with DNSSEC. There are four configuration files for the basic OpenDNSSEC installation:

- 591 ■ **conf.xml** which is the overall configuration of the system
- 592 ■ **kasp.xml** which contains the policy of signing
- 593 ■ **zonelist.xml** where you list all the zones that you are going to sign

1. The NLnet Labs OpenDNSSEC team provided most of the text in this section. This text is also available in an expanded form on OpenDNSSEC Wiki <https://wiki.opendnssec.org/display/DOCS20/OpenDNSSEC>.

594 ■ **addns.xml** (per zone, optional) for zone transfers

595 For now, it is necessary to edit **conf.xml** only because we need to configure the cryptographic
 596 security module must be configured (e.g., an HSM or software module such as SoftHSM or
 597 SoftHSM 2.x). Make the **Repository** part look like:

```
598 <Repository name="SoftHSM">  
599   <Module>/usr/local/lib/libsofthsm.so</Module>  
600   <TokenLabel>OpenDNSSEC</TokenLabel>  
601   <PIN>XXXX</PIN>  
602   <SkipPublicKey/>  
603 </Repository>
```

604 Here, **XXXX** is the user PIN entered in [section 2.4.1](#) above.

605 OpenDNSSECs Key and Signing Policy (KASP) provides standard values for signing any zone.
 606 However, if an organization chooses to change any value, it is possible to add a new policy, or
 607 change values in an existing policy. For example, if a zone uses the **YYYYMMDDXX** format for
 608 **SOA SERIAL** values, change the **Serial** parameter in **kasp.xml** from **unixtime** to **datecounter**:

```
609 <Zone>  
610   <PropagationDelay>PT9999S</PropagationDelay>  
611   <SOA>  
612     <TTL>PT3600S</TTL>  
613     <Minimum>PT3600S</Minimum>  
614     <Serial>datecounter</Serial>  
615   </SOA>  
616 </Zone>
```

617 For full descriptions about all the KASP parameters, see the OpenDNSSEC Wiki¹.

618 2.5.4 Running OpenDNSSEC

619 When starting OpenDNSSEC for the first time, it is first necessary to setup the database. There
 620 is a control script that starts up two daemons: **ods-enforcerd** that takes care of the key
 621 management, and **ods-signerd** that is the actual signer.

622 Run:

```
623 $ ods-enforcer-db-setup  
624 *WARNING* This will erase all data in the database; are you sure? [y/n]  
625 Y  
626 $ ods-control start
```

627 At this point, OpenDNSSEC is running. Logs are going to syslog. The setup has imported the two
 628 default Key And Signing Policies (KASP), **default** and **lab**. However, no zones are imported yet.

1. OpenDNSSEC Documentation: <https://wiki.opendnssec.org/display/DOCS20/kasp.xml>.

629 2.5.5 Adding Zones

630 Until the zone list **zonelist.xml** is edited, OpenDNSSEC starts with no zones to sign. It is
 631 necessary to add zones (and remove zones as necessary). One way to add a zone is to enter the
 632 following command:

```
633 $ ods-enforcer zone add -z example.com
```

634 This adds the zone **example.com** to OpenDNSSEC with the default KASP. Also by default, the
 635 signing is file based. Note that the enforcer doesn't read this file without being told explicitly to
 636 do so. Also, the file will not be written when adding new zones via **commandline**.

637 The signer expects the unsigned file to be at `/var/opendnssec/unsigned/example.com` and
 638 puts the signed file at `/var/opendnssec/signed/example.com`. Different paths can be
 639 used with **-i** (input) and **-o** (output). You can use a different policy with **-p** (policy).

640 If a user or administrator wants to use DNS zone transfers for input and output, the type of
 641 adapter can be set to DNS, **-j** for input and **-q** for output. It is necessary to set the input and
 642 output files to the zone transfer configuration file **addns.xml**, like this:

```
643 $ ods-ksmutil zone add -z example.com -j DNS -q DNS \
  644   -i /etc/opendnssec/addns.xml -o /etc/opendnssec/addns.xml
```

645 Instructions on how to edit **addns.xml** for zone transfers is described in [section 2.5.5.1](#) below.

646 The signed zone is then written in the `/var/opendnssec/signed/` directory. It is necessary
 647 to notify your name server of the new **zonefile** in order for the zone to also become visible in
 648 the DNS. It is possible to configure a **notify** command in **conf.xml** to automatically notify the
 649 name server of new zones. For example:

```
650 <Configuration>
  651   ...
  652   <Signer>
  653     ...
  654   <NotifyCommand>nameserver_control_program reload
  655     %zone</NotifyCommand>
  656   </Signer>
  657 </Configuration>
```

658 Here, **%zone** will be replaced with the name of the zone that has been updated, and **%zonefile**
 659 (not used in example) will be replaced with the name of the signed zonefile.

660 2.5.5.1 OpenDNSSEC as a Bump-in-the-Wire

661 If a zone has been added with DNS adapters rather than working on files, instead of pointing
 662 the input and output to the filenames of the unsigned and signed zones, it is necessary to put in
 663 the zone transfer configuration file **addns.xml**. Here, primary name server addresses, ports and
 664 TSIG keys (Inbound), and ports and TSIG keys for the secondary name servers (Outbound) are
 665 set up. Replace the example values in **addns.xml.sample** installed in `/etc/opendnssec/` with
 666 the desired servers and keys and rename it to **addns.xml**. Also **conf.xml** needs a socket that
 667 listens to DNS traffic:

```
668 <Configuration>
  669   ...
```

```

670      <Signer>
671      ...
672      <Listener>
673          <Interface><Address>127.0.0.1</Address><Port>53</Port></Interface>
674          <Interface><Address>::1</Address><Port>53</Port></Interface>
675      <Listener>
676          </Signer>
677      </Configuration>

```

678 The above values are also the defaults. OpenDNSSEC can now sign incoming zone transfers (full
679 and incremental) and also reply to SOA, AXFR and IXFR requests.

680 2.5.5.2 Activating Key Signing Keys (KSK)

681 At this stage, an attempt to list OpenDNSSEC keys will reveal that the key signing key (KSK) is not
682 yet active:

```

683 $ ods-enforcer key list -a
684 Zone: Keytype: State: Date of next transition:
685 example.com. KSK publish 2016-09-01 00:00:01 example.com. ZSK active
686 2016-08-31 10:00:01

```

687 This is because the DS must still be submitted to the parent. The DS is a record that is derived
688 from the KSK and is published in the parent zone. This is used to build a secure chain of trust
689 from the root zone to the users zone. In the example above, OpenDNSSEC expects this to
690 happen at one second past midnight on the first of September 2016. This is 14 hours after initial
691 signing. This is because the default policy has a very conservative propagation delay for the
692 name servers: 12 hours. In this example, it takes an additional hour for the **TTL** and one more
693 for the publish safety parameter - totaling 14 hours. Enduring the long propagation delay is
694 necessary because, in order to make sure a zone remains valid, it is necessary to respect a
695 publish safety duration and the **TTL** (in this case derived from the **SOA MINIMUM**). If
696 OpenDNSSEC is ready, the date of next transition be displayed as **waiting for ds-seen**. The DS
697 can then be submitted to the parent. How that is accomplished depends on your organization's
698 registrar. Usually this can be done via e-mail or through a web interface. Retrieve the DNSKEY or
699 DS with:

```

700 $ ods-enforcer key export
701 ;ready KSK DNSKEY record: example.com. 3600 IN DNSKEY 257 3 8 Aw...
702 $ ods-enforcer key export -d
703 ;ready KSK DS record (SHA1): example.com.. 3600 IN DS 42112 8 1 8aea...
704 ;ready KSK DS record (SHA256): example.com. 3600 IN DS 42112 8 2
705 a674...

```

706 If the DS shows up in the parent zone at all parent name servers, it is safe to run the **key**
707 **ds-seen** command. This command requires the keytag of the key in question. You can see from
708 the DNSKEY and DS records this is 42112 in this example:

```
709 $ ods-enforcer key ds-seen -z example.com -x 42112
```

710 The KSK is now also active, and the chain-of-trust is set up.

711 2.6 Unbound

712 The log file for an actual NCCoE installation and configuration of Unbound with NSD4 and
 713 OpenDNSSEC for the DNS-Based Email Security project is provided as [Appendix F](#). The latest
 714 version of unbound (currently 1.5.10) can always be downloaded from
 715 <http://www.unbound.net/downloads/unbound-latest.tar.gz>.¹ Unbound documentation can be
 716 found at <https://unbound.net/documentation/index.html>. Some general installation and
 717 configuration information for Unbound is provided in the following subsections. Please note
 718 that IP addresses, domain names, and mail addresses are, in many cases, specific to the NCCoE
 719 laboratory configuration and must not be used in actual implementations.

720 2.6.1 Unbound Installation Basics and System Requirements

721 If your distribution package manager includes a package for Unbound install the package with
 722 the package manager. If not, in order to compile the software it is necessary to have **openssl**,
 723 and its include files (from a package often called **openssl-devel**). In **openssl**, run **./configure**
 724 [**options**]; **make**; and **make install**. For cases in which the **libldns** library is not installed, a
 725 version is included with the Unbound source **tarball** and is automatically used. Unbound always
 726 uses **sldns** (the included **ldns**). With respect to options for **configure**, the default **config**
 727 locations for various files and directories can be customized, as well as the install location for
 728 the program with **--prefix=/usr/local**. You can specify **--with-libevent=dir** or
 729 **--with-ssl=dir** to link with the library at that location. In general, no options are needed for
 730 **./configure**.

731 On some BSD systems it is necessary to use **gmake** instead of **make**.

732 It is possible to install with **make install** and to uninstall with **make uninstall**. The uninstall
 733 does not remove the **config** file. In the **contrib directory** in the unbound source are sample **rc.d**
 734 scripts for unbound (for BSD and Linux type systems).

735 2.6.2 Unbound Setup and Installation

736 The **config** file is copied into **/usr/local/etc/unbound/unbound.conf** but some
 737 distributions may put it in **/etc/unbound/unbound.conf** or **/etc/unbound.conf**. The
 738 **config** file is fully annotated, you can go through it and select the options you like. Or you can
 739 use the below, a quick set of common options to serve the local subnet. A common setup for
 740 DNS service for an IPv4 subnet and IPv6 *localhost* is below. You can change the IPv4 subnet to
 741 match the subnet that you use, and add your IPv6 subnet if you have one.

```
742 # unbound.conf for a local subnet.
743 server:
744 interface: 0.0.0.0
745 interface: ::0
```

1.Source: unbound-1.5.9.tar.gz; SHA1 checksum: 4882c52aac0ab-
 cd72a86ac5d06e9cd39576620ce; SHA256 checksum: 01328cfac99ab5b8c47115151896a244979e442e284eb962c0ea84b7782b6990; PGP signature:
 unbound-1.5.9.tar.gz.asc; License: BSD; Doc: man-page.

```
746 access-control: 192.168.0.0/16 allow
747 access-control: ::1 allow
748 verbosity: 1
```

749 By default the software comes with **chroot** enabled. This provides an extra layer of defense
 750 against remote exploits. Enter file paths as full pathnames starting at the root of the filesystem
 751 ('/'). If **chroot** gives you trouble, you can disable it with **chroot: ""** in the **config**. Also the
 752 server assumes the username **unbound** to drop privileges. You can add this user with your
 753 favorite account management tool (*useradd(8)*), or disable the feature¹ with **username: ""** in
 754 the **config**.

755 Start the server using the script (if you or the package manager installed one) as
 756 **/etc/rc.d/init.d/unbound start**. Or **unbound -c <config>** as root.

757 It is possible to setup remote control using **unbound-control**. First run
 758 **unbound-control-setup** to generate the necessary TLS key files (they are put in the default
 759 install directory). If you use a username of **unbound** to run the daemon from use **sudo -u**
 760 **unbound unbound-control-setup** to generate the keys, so that the server is allowed to read
 761 the keys. Then add the following at the end of the config file:

```
762 # enable remote-control
763 remote-control:
764 control-enable: yes
```

765 You can now use **unbound-control** to send commands to the daemon. It needs to read the key
 766 files, so you may need to **sudo unbound-control**. Only connections from *localhost* are allowed
 767 by default

768 2.6.3 Unbound Configuration for DNSSEC

769 DNSSEC is a mechanism to protect DNS data. It uses digital signatures. To use DNSSEC with
 770 Unbound, the public keys for digital signature must be configured. Note that specific
 771 distributions, operating systems, or device vendors may have already provided the anchor,
 772 securing it with its own vendor-specific update mechanism. In that case, the mechanisms
 773 provided from those sources should be used.

774 2.6.3.1 Trust Anchor

775 The first step in configuring Unbound for DNSSEC is to obtain an initial trust anchor.² The
 776 **unbound-anchor** tool provides an initial anchor from built-in values, but for real trust this
 777 should be checked thoroughly. The root key is stored in a file,
 778 **/usr/local/etc/unbound/root.key**. Unbound must be able to read and write it, to keep
 779 it up to date with the latest key(s). It must therefore reside within the **chroot** of Unbound (if
 780 that is used). Access rights are world-readable, user Unbound write only. Use **sudo -u unbound**
 781 to start **unbound-anchor** so that the file owner is set to the unbound user (same username as
 782 daemon uses). It can optionally be put somewhere else, accessible to the unbound daemon,
 783 such as **/var/unbound** or **/etc**. You need to pass this value to **unbound-anchor** (option **-a**

1. Do not run as **root**.

2. Unbound: How to enable DNSSEC, W.C.A. Wijngaards, NLnet Labs, April 2011.

784 file) and to unbound (**auto-trust-anchor-file**: “file” in **unbound.conf**). The **unbound-anchor**
 785 tool creates this file for the administrator if it does not exist. But the administrator must check
 786 this file so that it can be trusted. The **unbound-anchor** tool also has a built-in certificate (from
 787 the ICANN Certificate Authority) that it will use to update the root key if it becomes out of date,
 788 this should be checked too (-1 option to show it), or provide some other certificate that
 789 **unbound-anchor** is to use.

790 There are trusted community representatives that have sworn and signed attestations, and
 791 there may be publications (i.e. in printed form). Please notice that NLnet Labs' **unbound-anchor**
 792 tool provides an initial value for convenience, systems administrators must perform the
 793 specified checks to obtain trust. The trust anchor can be downloaded via https from IANA:
 794 [root-anchors.xml](https://www.iana.org/assignments/root-anchors/root-anchors.xml) (click link and then check the lock icon and the *urlbar* and the hash displayed
 795 against the hash you can put as initial value into the **root.key** file, see below for an example of
 796 the syntax of how to input the initial value).

797 Here is the 2010-2011 trust anchor for the root zone. This is the syntax that you can use to
 798 provide an initial value for the **root.key** file:

799 . IN DS 19036 8 2
 800 49AAC11D7B6F6446702E54A1607371607A1A41855200FD2CE1CDDE32F24E8FB5

801 2.6.3.2 Update Mechanism Setup

802 Set the **unbound-anchor** tool to run at system startup, it is part of the Unbound package. A
 803 good way is to run it from the **init** scripts, with **sudo -u unbound** so that the file permissions
 804 work out.

805 Before **unbound-anchor** is run inside the **init** scripts, you must run **NTP** (in secure mode), so
 806 that the time and date have been set properly. Unbound uses RFC5011 updates to keep the
 807 anchor updated if it is changed while the computer is in operation, but the **unbound-anchor**
 808 tool is used if it is changed while the computer is not in operation.

809 In the **unbound.conf** config file, include the root anchor file with the automatic updated anchor
 810 statement, like this:

```
811 server:  

  812     # ... other stuff  

  813     # root key file, automatically updated  

  814     auto-trust-anchor-file: "/usr/local/etc/unbound/root.key"
```

815 After you change the config, restart unbound. Unbound will then overwrite the key file with
 816 status information (such as the last time the key was seen).

817 2.6.3.3 Testing Unbound Configurations for DNSSEC

818 Entering **dig com. SOA +dnssec** should result in display of the AD flag there. If this is
 819 unsuccessful, the Unbound **option val-log-level: 2** should log explanations regarding why
 820 the DNSSEC validation fails (one line per failed query). Also, <http://test.dnssec-or-not.org/> (fun
 821 test) or <https://internet.nl/> (sober test) and <http://www.kaminskybug.se/> (look for a happy bug
 822 icon) are useful test tools.

2.6.4 Unbound Support

824 Although it is open source software, support for Unbound is available from a number of
825 sources, including NLnet Labs.

2.7 How to Install and Configure a DNS Signer Platform

827 DNS Signer is a commercial product, the installation and configuration instructions can be
828 obtained from the company website, <http://www.secure64.com/>.

2.7.1 DNS Signer Installation Basics and System Requirements

830 Secure64 DNS Signer runs on HP Integrity servers with the following minimum configuration:

- 831 ■ 1 dual core Itanium microprocessor
- 832 ■ 4 GB RAM
- 833 ■ 36 GB disk drive
- 834 ■ DVD ROM drive

835 DNS Signer is a commercial product. Information regarding obtaining the product can be found
836 at <http://www.secure64.com/contact>.

2.7.2 DNS Signer Installation and Configuration

837 DNS Signer can be configured to work with an authoritative DNS resolver, (e.g., DNS Authority)
839 or a caching/recursive resolver (e.g., DNS Cache). The process followed for installation of DNS
840 Signer at the NCCoE is included in [Appendix H](#).

2.8 How to Install and Configure a DNS Authority Platform

841 DNS Authority is a commercial product, the installation and configuration instructions can be
842 obtained from the company website, <http://www.secure64.com/>. Information regarding
843 obtaining the product can be found at <http://www.secure64.com/contact>. DNS Authority can
844 be configured to work with a caching/recursive resolver (e.g., DNS Cache) and a DNS Signer. The
845 process followed for installation of DNS Authority at the NCCoE is included in Appendix H.
846

2.9 How to Install and Configure DNS Cache

848 DNS Cache is a commercial product, installation and configuration instructions can be obtained
849 from the company website, <http://www.secure64.com/>. Information regarding obtaining the
850 product can be found at <http://www.secure64.com/contact>.

2.10 How to Install and Configure a Dovecot/Postfix Mail Transfer Agent

2.10.1 Dovecot Installation Basics and System Requirements

Dovecot can be downloaded from sources identified at the Dovecot Secure IMAP Server site (<http://www.dovecot.org/download.html>).

2.10.1.1 Compiling Dovecot from Source Code

To compile Dovecot from source code provide the following commands:

```
./configure
make
sudo make install
```

That installs Dovecot under the `/usr/local` directory. The configuration file is in `/usr/local/etc/dovecot.conf`. Logging goes to syslog's mail facility by default, which typically goes to `/var/log/mail.log` or something similar. If you are in a hurry, you can then jump to QuickConfiguration. If you have installed some libraries into locations which require special include or library paths, you can pass them in the **CPPFLAGS** and **LDFLAGS** environment variables. For example:

```
CPPFLAGS="-I/opt/openssl/include" LDFLAGS="-L/opt/openssl/lib"
./configure
```

It is necessary to create two users for Dovecot's internal use:

- **dovecnull**: Used by untrusted imap-login and pop3-login processes (`default_login_user` setting).
- **dovecot**: Used by slightly more trusted Dovecot processes (`default_internal_user` setting).

Each of them should also have its own **dovecnull** and **dovecot** groups. See <http://wiki2.dovecot.org/UserIds> for more information.

2.10.1.2 Compiling Dovecot from Git

Dovecot is available from Git, for example with:

```
git clone https://github.com/dovecot/core.git dovecot
```

To compile Dovecot from Git, it is first necessary to run `./autogen.sh` to generate the configure script and some other files. This requires that the following software/packages be installed:

- autoconf
- automake
- libtool
- pkg-config
- gettext

887 ■ GNU make

888 It is advisable to add **--enable-maintainer-mode** to the configure script:

889 ./autogen.sh

890 ./configure --enable-maintainer-mode

891 make

892 sudo make install

893 For later updates, the commands are:

894 git pull

895 make

896 sudo make install

897 2.10.1.3 Compiling Dovecot with rpmbuild (Mandriva, RedHat, etc.)

898 Fetch the source rpm from <ftp://ftp.surfnet.nl/> or any other mirror. Currently,
 899 **dovecot-10.rc26.src.rpm** can be found in the **cooker** subtree. If the current release is newer,
 900 unpack the source rpm with **rpm -ivh dovecot-10.rc26.src.rpm** to a build environment
 901 (/usr/src/rpm...) Copy the newer **tarball** from the dovecot site to the **SOURCES** directory
 902 of the build environment. Change the **dovecot.spec** file in the **SPECS** directory to reflect the
 903 new release and the new name of the **tarball**. The maintainer works with a **bz2 tarball**; a **tar.gz**
 904 **tarball** makes no difference. Issue a **rpmbuild -ba dovecot.spec**. The resulting rpm will be
 905 placed in **RPMS/i586**. Install with **rpm** or **urpmi**:

906 rpm -ivh dovecot-1.0.rc26.src.rpm
 907 cd /usr/src/rpm
 908 mv ~/downloads/dovecot-1.0.rc28.tar.gz ./SOURCES
 909 cd SPECS
 910 vi dovecot.spec
 911 ...edit release and tarball name. Change default options if needed...
 912 rpmbuild -ba dovecot.spec
 913 cd ../RPMS/i586
 914 urpmi ./dovecot-1.0.rc28-1mdv2007.0.i586.rpm

915 During this process missing prerequisites may be detected. Install them and rerun the build
 916 process. The spec file also need updating for the new add-ons (**idxview** and **logview**).

917 2.10.1.4 SSL/TLS Support

918 Dovecot was initially built to support both OpenSSL and GNUTLS, but OpenSSL is currently used
 919 by default, and it should be automatically detected. If it is not, some header files or libraries are
 920 missing, or they are in a non-standard path. The **openssl-dev** or a similar package needs to be
 921 installed, and if it is not in the standard location, set **CPPFLAGS** and **LDLIBS** as shown above.
 922 By default the SSL certificate is read from **/etc/ssl/certs/dovecot.pem**, and the private
 923 key from **/etc/ssl/private/dovecot.pem**. The **/etc/ssl** directory can be changed using
 924 the **--with-ssldir=DIR** configure option. Both can of course be overridden from the
 925 configuration file.

```

926 For Linux installations, note that current inotify is in the Linux kernel since version 2.6.13 and it
927 is preferred over dnotify. If your distribution does not have the required inotify header file, it
928 can be obtained from the inotify maintainer (the following example requires cURL):
929
930     mkdir -p /usr/local/include/sys
931     cd /usr/local/include/sys
932     curl
933     ftp://ftp.kernel.org/pub/linux/kernel/people/rml/inotify/headers/inoti
934     fy.h -O
935     curl
936     ftp://ftp.kernel.org/pub/linux/kernel/people/rml/inotify/headers/inoti
937     fy-syscalls.h >> inotify.h
938
939     /usr/local/include isn't in standard include lookup path, so that needs to be specified to
940     configure:
941
942     CPPFLAGS=-I/usr/local/include ./configure --with-notify=inotify

```

940 2.10.1.5 Dovecot Configuration Options

- 941 ■ **--help**
942 gives a full list of available options
- 943 ■ **--help=short**
944 just lists the options added by the particular package (= Dovecot)

945 Options are usually listed as **--with-something** OR **--enable-something**. If you want to disable
946 them, do it as **--without-something** OR **--disable-something**. There are many default
947 options that come from autoconf, automake or libtool. The list of options that Dovecot adds
948 follows:

- 949 ■ **--enable-devel-checks**
950 Enables some extra sanity checks. This is mainly useful for developers. It does quite a lot of
951 unnecessary work but should catch some programming mistakes more quickly.
- 952 ■ **--enable-asserts**
953 Enable assertion checks, enabled by default. Disabling them may slightly save some CPU,
954 but if there are bugs they can cause more problems since they are not detected as early.
- 955 ■ **--without-shared-libs**
956 Link Dovecot binaries with static libraries instead of dynamic libraries.
- 957 ■ **--disable-largefile**
958 Specifies if we use 32bit or 64bit file offsets in 32bit CPUs. 64bit is the default if the system
959 supports it (Linux and Solaris do). Dropping this to 32bit may save some memory, but it
960 prevents accessing any file larger than 2 GB.
- 961 ■ **--with-mem-align=BYTES**
962 Specifies memory alignment used for memory allocations. It is needed with many non-x86
963 systems and it should speed up x86 systems too. Default is 8, to make sure 64bit memory
964 accessing works.
- 965 ■ **--with-ioloop=ILOOP**

966 Specifies what I/O loop method to use. Possibilities are select, poll, epoll and kqueue. The
967 default is to use the best method available on your system.

968 ■ **--with-notify=NOTIFY**
969 Specifies what file system notification method to use. Possibilities are dnotify, inotify (both
970 on Linux), kqueue (FreeBSD) and none. The default is to use the best method available on
971 your system. See Notify method above for more information.

972 ■ **--with-storages=FORMATS**
973 Specifies what mailbox formats to support. Note: Independent of this option, the formats
974 raw and shared will be always built.

975 ■ **--with-solr**
976 Build with Solr full text search support

977 ■ **--with-zlib**
978 Build with zlib compression support (default if detected)

979 ■ **--with-bzlib**
980 Build with bzip2 compression support (default if detected)

981 SQL Driver Options

982 SQL drivers are typically used only for authentication, but they may be used as a lib-dict
983 backend too, which can be used by plugins for different purposes.

984 ■ **--with-sql-drivers**
985 Build with specified SQL drivers. Defaults to all that were found with autodetection.

986 ■ **--with-pgsql**
987 Build with PostgreSQL support (requires pgsql-devel, libpq-dev or similar package)

988 ■ **--with-mysql**
989 Build with MySQL support (requires mysql-devel, libmysqlclient15-dev or similar package)

990 ■ **--with-sqlite**
991 Build with SQLite3 driver support (requires sqlite-devel, libsqlite3-dev or similar package)

992 Authentication Backend Options

993 The basic backends are built if the system is detected to support them:

994 ■ **--with-shadow**
995 Build with shadow password support

996 ■ **--with-pam**
997 Build with PAM support

998 ■ **--with-nss**
999 Build with NSS support

1000 ■ **--with-sia**
1001 Build with Tru64 SIA support

1002 ■ **--with-bsdauth**
1003 Build with BSD authentication support (if supported by your OS)

1004 Some backends require extra libraries and are not necessarily wanted, so they are built only if
 1005 specifically enabled:

- 1006 ■ **--with-sql**
 1007 Build with generic SQL support (drivers are enabled separately)
- 1008 ■ **--with-ldap**
 1009 Build with LDAP support (requires openldap-devel, libldap2-dev or similar package)
- 1010 ■ **--with-gssapi**
 1011 Build with GSSAPI authentication support (requires krb5-devel, libkrb5-dev or similar
 1012 package)
- 1013 ■ **--with-vpopmail**
 1014 Build with vpopmail support (requires vpopmail sources or a development package)

1015 It's also possible to build these as plugins by giving e.g. **--with-sql=plugin**.

1016 **2.10.1.6 Dovecot Support**

1017 Although Dovecot is open source software, support is available from dovecot.org and
 1018 commercial sources. See <http://www.dovecot.org/support.html>.

1019 **2.10.2 Postfix Installation and Configuration**

1020 Postfix was released under the IBM Public License, and source code can be downloaded from
 1021 <http://cdn.postfix.johnriley.me/mirrors/postfix-release/index.html>. All Postfix source code is
 1022 signed with Wietse's PGP key.¹ Instructions for installing Postfix from source code can be found
 1023 at <http://www.postfix.org/INSTALL.html>. Postfix manual pages can be found at
 1024 <http://www.postfix.org/postfix-manuals.html>.

1025 **2.10.2.1 Installation and System Requirements**

1026 If you are using a pre-compiled version of Postfix, you should start with
 1027 `BASIC_CONFIGURATION_README` and the general documentation referenced by it. `INSTALL` is
 1028 only a bootstrap document to get Postfix up and running from scratch with the minimal number
 1029 of steps; it is not considered part of the general documentation. The `INSTALL` document
 1030 describes how to build, install and configure a Postfix system so that it can do one of the
 1031 following:

- 1032 ■ Send mail only, without changing an existing Sendmail installation.
- 1033 ■ Send and receive mail via a virtual host interface, still without any change to an existing
 1034 Sendmail installation.
- 1035 ■ Run Postfix instead of Sendmail.

1. See <http://ftp.porcupine.org/mirrors/project-history/postfix/> for a more extensive archive of
 tarballs.

1036 According to INSTALL, Postfix development is conducted on FreeBSD and MacOS X, with regular
1037 tests on Linux (Fedora, Ubuntu) and Solaris. Support for other systems relies on feedback from
1038 their users, and may not always be up-to-date. OpenBSD is partially supported. The libc resolver
1039 does not implement the documented "internal resolver options which are [...] set by changing
1040 fields in the **_res structure**" (documented in the OpenBSD 5.6 resolver(3) manpage). This results
1041 in too many DNS queries, and false positives for queries that should fail.

1042 2.10.2.2 Compiler Specifics

1043 If you need to build Postfix for multiple architectures from a single source-code tree, use the
1044 **lndir** command to build a shadow tree with symbolic links to the source files. If at any time in
1045 the build process you get messages like: `make: don't know how to ...` you should be
1046 able to recover by running the following command from the Postfix top-level directory:

1047 `$ make -f Makefile.init makefiles`

1048 If you copied the Postfix source code after building it on another machine, it is a good idea to `cd`
1049 into the top-level directory and first do this:

1050 `$ make tidy`

1051 This will get rid of any system dependencies left over from compiling the software elsewhere.

1052 To build with GCC, or with the native compiler if people told me that is better for your system,
1053 just `cd` into the top-level Postfix directory of the source tree and type:

1054 `$ make`

1055 To build with a non-default compiler, you need to specify the name of the compiler, for
1056 example:

1057 `$ make makefiles CC=/opt/SUNWspro/bin/cc (Solaris)`

1058 `$ make`

1059 `$ make makefiles CC="/opt/ansic/bin/cc -Ae (HP-UX)`

1060 `$ make`

1061 `$ make makefiles CC="purify cc"`

1062 `$ make`

1063 In some cases, optimization will be turned off automatically.

1064 2.10.2.3 Building with Position-Independent Executables

1065 On some systems Postfix can be built with Position-Independent Executables. PIE is used by the
1066 ASLR exploit mitigation technique (ASLR = Address-Space Layout Randomization).

1067 `$ make makefiles pie=yes ...other arguments...`

1068 (Specify `make makefiles pie=no` to explicitly disable Postfix position-independent executable
1069 support). Postfix PIE support appears to work on Fedora Core 20, Ubuntu 14.04, FreeBSD 9 and
1070 10, and NetBSD 6 (all with the default system compilers). Whether the `pie=yes` above has any
1071 effect depends on the compiler. Some compilers always produce PIE executables, and some
1072 may even complain that the Postfix build option is redundant.

1073 **2.10.2.4 Dynamically Linked Libraries**

1074 Postfix dynamically-linked library and database plugin support exists for recent versions of
 1075 Linux, FreeBSD and MacOS X. Note that dynamically-linked library builds may become the
 1076 default at some point in the future.

1077 **2.10.2.5 Default Settings and Optional Features**

1078 By default, Postfix builds as a mail system with relatively few bells and whistles. Support for
 1079 third-party databases etc. must be configured when Postfix is compiled. The following
 1080 documents describe how to build Postfix with support for optional features:

1081 **Table 2.5 Postfix Default Settings and Optional Features**

Optional Feature	Document	Availability
Berkeley DB database	DB_README	Postfix 1.0
LMDB database	LMDB_README	Postfix 2.11
LDAP database	LDAP_README	Postfix 1.0
MySQL database	MYSQL_README	Postfix 1.0
Perl compatible regular expression	PCRE_README	Postfix 1.0
PostgreSQL database	PGSQL_README	Postfix 2.0
SASL authentication	SASL_README	Postfix 1.0
SQLite database	SQLITE_README	Postfix 2.8
STARTTLS session encryption	TLS_README	Postfix 2.2

1082 Note: *IP version 6 support is compiled into Postfix on operating systems that have IPv6 support.*
 1083 See the [IPV6_README](#) file for details.

1084 **2.10.2.6 Installing After Compiling**1085 **1. Save existing Sendmail binaries**

1086 Some systems implement a mail switch mechanism where different MTAs (Postfix,
 1087 Sendmail, etc.) can be installed at the same time, while only one of them is actually being
 1088 used. Examples of such switching mechanisms are the FreeBSD mailwrapper(8) or the Linux
 1089 mail switch. In this case you should try to “flip” the switch to “Postfix” before installing
 1090 Postfix. If your system has no mail switch mechanism, execute the following commands
 1091 (your sendmail, newaliases and mailq programs may be in a different place):

```
1092 ?# mv /usr/sbin/sendmail /usr/sbin/sendmail.OFF
1093 # mv /usr/bin/newaliases /usr/bin/newaliases.OFF
1094 # mv /usr/bin/mailq /usr/bin/mailq.OFF
1095 # chmod 755 /usr/sbin/sendmail.OFF/usr/bin/newaliases.OFF\
1096 /usr/bin/mailq.OFF
```

1097 **2. Create account and groups**

1098 Before you install Postfix for the first time you need to create an account and a group:

1099 a. Create a user account **postfix** with a user id and group id that are not used by any other
1100 user account. Preferably, this is an account that no-one can log into. The account does
1101 not need an executable login shell, and needs no existing home directory. Sample
1102 password and group file entries follow:

1103 /etc/passwd:
1104 postfix:*:12345:12345:postfix:/no/where:/no/shell
1105 /etc/group:
1106 postfix:*:12345:
1107 Note: there should be no whitespace before **postfix**:.

1108 b. Create a group **postdrop** with a group id that is not used by any other user account. Not
1109 even by the postfix user account. An example of a group file entry follows:

1110 /etc/group:
1111 postdrop:*:54321:
1112 Note: there should be no whitespace before **postdrop**:.

1113 3. Install Postfix

1114 To install or upgrade Postfix from compiled source code, run one of the following
1115 commands as the super-user:

1116 # make install (interactive version, first time install)
1117 # make upgrade (non-interactive version, for upgrades)

1118 a. The interactive version (**make install**) asks for pathnames for Postfix data and
1119 program files, and stores your preferences in the **main.cf** file. If you don't want Postfix
1120 to overwrite non-Postfix **sendmail**, **mailq** and **newaliases** files, specify pathnames that
1121 end in **.postfix**.

1122 b. The non-interactive version (**make upgrade**) needs the **/etc/postfix/main.cf** file
1123 from a previous installation. If the file does not exist, use interactive installation (**make**
1124 **install**) instead.

1125 If you specify **name=value** arguments on the **make install** or **make upgrade** command
1126 line, then these will take precedence over compiled-in default settings or **main.cf**
1127 settings. The command **make install/upgrade name=value ...** will replace the
1128 string **MAIL_VERSION** at the end of a configuration parameter value with the Postfix
1129 release version. Do not try to specify something like **\$mail_version** on this command
1130 line. This produces inconsistent results with different versions of the **make(1)**
1131 command.

1132 2.10.2.7 Configure Postfix

1133 See <http://www.postfix.org/postconf.5.html> for Postfix configuration parameters.

1134 Note: The material covered in this section from **INSTALL** Section 10 is covered in more detail in
1135 the **BASIC_CONFIGURATION_README** document. The information presented below is
1136 targeted at experienced system administrators.

1137 1. Postfix configuration files

1138 By default, Postfix configuration files are in `/etc/postfix`. The two most important files
 1139 are **main.cf** and **master.cf**; these files must be owned by root. Giving someone else write
 1140 permission to **main.cf** or **master.cf** (or to their parent directories) means giving root
 1141 privileges to that person. In `/etc/postfix/main.cf`, you will have to set up a minimal
 1142 number of configuration parameters. Postfix configuration parameters resemble shell
 1143 variables, with two important differences: the first one is that Postfix does not know about
 1144 quotes like the UNIX shell does. You specify a configuration parameter as:

```
1145 /etc/postfix/main.cf:  
1146     parameter = value
```

1147 and you use it by putting a "\$" character in front of its name:

```
1148 /etc/postfix/main.cf:  
1149     other_parameter = $parameter
```

1150 You can use **\$parameter** before it is given a value (that is the second main difference with
 1151 UNIX shell variables). The Postfix configuration language uses lazy evaluation, and does not
 1152 look at a parameter value until it is needed at runtime. Whenever you make a change to the
 1153 **main.cf** or **master.cf** file, execute the following command in order to refresh a running mail
 1154 system:

```
1155     # postfix reload
```

1156 2. Default domain for unqualified addresses

1157 First of all, you must specify what domain will be appended to an unqualified address (i.e.
 1158 an address without **@domain.tld**). The **myorigin** parameter defaults to the local hostname,
 1159 but that is intended only for very small sites.

1160 Some examples (use only one) :

```
1161 /etc/postfix/main.cf:  
1162     myorigin = $myhostname (send mail as "user@$myhostname")  
1163     myorigin = $mydomain (send mail as "user@$mydomain")
```

1164 3. Specification of what domains to receive locally

1165 Next you need to specify what mail addresses Postfix should deliver locally.

1166 Some examples (use only one):

```
1167 /etc/postfix/main.cf:  
1168     mydestination = $myhostname, localhost.$mydomain, localhost  
1169     mydestination = $myhostname, localhost.$mydomain,  
1170     localhost,$mydomain  
1171     mydestination = $myhostname
```

1172 The first example is appropriate for a workstation, the second is appropriate for the mail
 1173 server for an entire domain. The third example should be used when running on a virtual
 1174 host interface.

1175 4. Proxy/NAT interface addresses

1176 The **proxy_interfaces** parameter specifies all network addresses that Postfix receives mail
1177 on by way of a proxy or network address translation unit. You may specify symbolic
1178 hostnames instead of network addresses.

1179 **IMPORTANT:** You must specify your proxy/NAT external addresses when your system is a
1180 backup MX host for other domains, otherwise mail delivery loops will happen when the
1181 primary MX host is down.

1182 Example: host behind NAT box running a backup MX host.

```
1183     /etc/postfix/main.cf:  
1184         proxy_interfaces = 1.2.3.4 (the proxy/NAT external network address)
```

1185 5. Specification of What local clients to relay mail from

1186 If your machine is on an open network then you must specify what client IP addresses are
1187 authorized to relay their mail through your machine into the Internet. The default setting
1188 includes all subnetworks that the machine is attached to. This may give relay permission to
1189 too many clients. For example:

```
1190     /etc/postfix/main.cf:  
1191         mynetworks = 168.100.189.0/28, 127.0.0.0/8
```

1192 6. Specification of what relay destinations to accept from strangers

1193 If your machine is on an open network then you must also specify whether Postfix will
1194 forward mail from strangers. The default setting will forward mail to all domains (and
1195 subdomains of) what is listed in **\$mydestination**. This may give relay permission for too
1196 many destinations. Recommended settings (use only one):

```
1197     /etc/postfix/main.cf:  
1198         relay_domains = (do not forward mail from strangers)  
1199         relay_domains = $mydomain (my domain and subdomains)  
1200         relay_domains = $mydomain, other.domain.tld, ...
```

1201 7. Optional: configure a smart host for remote delivery

1202 If you're behind a firewall, you should set up a **relayhost**. If you can, specify the
1203 organizational domain name so that Postfix can use DNS lookups, and so that it can fall back
1204 to a secondary MX host when the primary MX host is down. Otherwise just specify a
1205 hard-coded hostname. Some examples follow (use only one):

```
1206     /etc/postfix/main.cf:  
1207         relayhost = $mydomain  
1208         relayhost = [mail.$mydomain]
```

1209 The form enclosed with [] eliminates DNS MX lookups. By default, the SMTP client will do
1210 DNS lookups even when you specify a relay host. If your machine has no access to a DNS
1211 server, turn off SMTP client DNS lookups like this:

```
1212     /etc/postfix/main.cf:  
1213         disable_dns_lookups = yes
```

1214 The [STANDARD_CONFIGURATION_README](#) file has more hints and tips for firewalled
1215 and/or dial-up networks.

1216 8. Create the aliases database

1217 Postfix uses a Sendmail-compatible aliases(5) table to redirect mail for local(8) recipients.
 1218 Typically, this information is kept in two files: in a text file /etc/aliases and in an indexed file
 1219 /etc/aliases.db. The command `postconf alias_maps` will tell you the exact location
 1220 of the text file. First, be sure to update the text file with aliases for root, postmaster and
 1221 postfix that forward mail to a real person. Postfix has a sample aliases file
 1222 /etc/postfix/aliases that you can adapt to local conditions.

```
1223 /etc/aliases:  

1224     root: you  

1225     postmaster: root  

1226     postfix: root  

1227     bin: root  

1228     etcetera...
```

1229 Note: there should be no whitespace before the ":". Finally, build the indexed aliases file
 1230 with one of the following commands:

```
1231     # newaliases  

1232     # sendmail -bi
```

1233 9. Setting up chroot

1234 Postfix daemon processes can be configured (via `master.cf`) to run in a chroot jail. The
 1235 processes run at a fixed low privilege and with access only to the Postfix queue directories
 1236 (/var/spool/postfix). This provides a significant barrier against intrusion. Note that
 1237 this barrier is not impenetrable, but every little bit helps. With the exception of Postfix
 1238 daemons that deliver mail locally and/or that execute non-Postfix commands, every Postfix
 1239 daemon can run chrooted.

1240 Sites with high security requirements should consider to chroot all daemons that talk to the
 1241 network: the `smtp(8)` and `smtpd(8)` processes, and perhaps also the `lmtp(8)` client. The
 1242 default `/etc/postfix/master.cf` file specifies that no Postfix daemon runs chrooted. In
 1243 order to enable chroot operation, edit the file `/etc/postfix/master.cf`. Instructions
 1244 are in the file.

1245 Note also that a chrooted daemon resolves all filenames relative to the Postfix queue
 1246 directory (/var/spool/postfix). For successful use of a chroot jail, most UNIX systems
 1247 require you to bring in some files or device nodes. The `examples/chroot-setup` directory in
 1248 the source code distribution has a collection of scripts that help you set up Postfix chroot
 1249 environments on different operating systems.

1250 Additionally, you need to configure `syslogd` so that it listens on a socket inside the Postfix
 1251 queue directory. Examples for specific systems:

1252 FreeBSD:

```
1253     # mkdir -p /var/spool/postfix/var/run  

1254     # syslogd -l /var/spool/postfix/var/run/log
```

1255 Linux, OpenBSD:

```
1256      # mkdir -p /var/spool/postfix/dev
1257      # syslogd -a /var/spool/postfix/dev/log
```

1258 2.10.3 Postfix Installation and Configuration for use with Dovecot

1259 The following elements are necessary for setting up Postfix for Dovecot¹:

- 1260 ■ A domain such as **mydomain.com**
- 1261 ■ A hostname for your mail server such as **mail.mydomain.com**
- 1262 ■ An SSL certificate that is valid for **mail.mydomain.com**

1263 2.10.3.1 Setting up SSL Certificate

1264 For SSL, you need a certificate and a private key saved in a location such as
 1265 `/etc/ssl/certs/mailcert.pem` and the key is saved (e.g., in
 1266 `/etc/ssl/private/mail.key`). Make sure the key is only readable by the root user. How to
 1267 set up SSL certificates for your website and e-mail depends on your website structure and the
 1268 CA you use (self-signed, organizational (sub)-ca, or commercial ca for example). Creating a
 1269 self-signed test certificate is as easy as executing

```
1270 sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
1271 /etc/ssl/private/mail.key -out /etc/ssl/certs/mailcert.pem2
```

1272 and leaving the default values in by just hitting enter on all questions asked.

1273 Most CAs will require you to submit a certificate signing request. (CSR) You can generate one
 1274 like this:

```
1275 sudo openssl req -nodes -days 365 -newkey rsa:2048 -keyout
1276 /etc/ssl/private/mail.key -out mailcert.csr
```

1277 Fill in the information queried properly, like in this transcript: (Check with the CA you intend to
 1278 use on what information needs to be in the CSR)

1279 Specific instructions for acquisition of certificates from CAs can be obtained from the CA. An
 1280 example is provided at:

1281 <https://www.digitalocean.com/community/tutorials/how-to-set-up-a-postfix-e-mail-server-with-dovecot>.

1283 2.10.3.2 Setting up DNS

1284 You still have to set up the DNS with an a record that points to your mail server IP and an MX
 1285 record that points to the mail servers hostname. Instructions for the standard configuration for
 1286 Postfix can be found at http://www.postfix.org/STANDARD_CONFIGURATION_README.html.

1. See How To Set Up a Postfix E-Mail Server with Dovecot, DigitalOcean, November 14, 2013.

<https://www.digitalocean.com/community/tutorials/how-to-set-up-a-postfix-e-mail-server-with-dovecot>

2. Don't use this certificate in your system.

2.11 How to Install and Configure a Thunderbird Mail Client

The starting point for installing Thunderbird can be found at <https://support.mozilla.org/en-US/kb/installing-thunderbird>, and the initial step is to click on the icon designating the operating system on which Thunderbird is being installed (Windows, Mac, or Linux).

2.11.1 Thunderbird Installation Basics and System Requirements

System requirements for installing Thunderbird 45.2.0 on Windows, Mac, and Linux operating systems can be found at <https://www.mozilla.org/en-US/thunderbird/45.2.0/system-requirements/>.

2.11.2 Thunderbird Installation and Configuration on Windows

Instructions for installing Thunderbird in Windows environments can be found at <https://support.mozilla.org/en-US/kb/installing-thunderbird-windows>. Selecting **Download** will download Thunderbird on the disk image **Thunderbird 45.2.0.dmg**. After starting the process by clicking **Run**, the Mozilla Thunderbird Setup Wizard will be started. Closing all other applications before starting Setup will make it possible to update relevant system files without having to reboot the computer. After installation, double-clicking on the Thunderbird icon runs the program.

2.11.3 Thunderbird Installation and Configuration on Linux

Instructions for installing Thunderbird on Linux can be found at <https://support.mozilla.org/en-US/kb/installing-thunderbird-linux>. To install Thunderbird using the package manager, it is necessary to refer to the documentation of the Linux distribution you're using. Complete instructions for installing Thunderbird outside of package management may be available at a distribution support website (e.g., [Installing Thunderbird on Ubuntu](#)).

2.11.4 Thunderbird Installation and Configuration on Mac

Instructions for installing Thunderbird on Mac machines can be found at <https://support.mozilla.org/en-US/kb/installing-thunderbird-on-mac>. The Thunderbird download page automatically detects the platform and language on the computer accessing it. To download Thunderbird in a language other than the one suggested, click on **Other Systems & Languages** for the list of available editions. Click on the OS X installation of your choice to continue. Once the download is completed, the disk image may open by itself and mount a new volume which contains the Thunderbird application. If you do not see the new volume, double-click the Thunderbird **dmg** icon to open it. A Finder window appears, containing the Thunderbird application. Drag the Thunderbird icon to the Applications folder. At this point you can eject the disk image by selecting it in a Finder window and pressing the **command+E** keys or by using the Finder's File menu, and selecting Eject. Open the Applications folder and double-click on the Thunderbird icon to start it. You may get a security warning that

1324 Thunderbird has been downloaded from the Internet. Because you downloaded Thunderbird
1325 from the official site, you can click **Open** to continue. The first time you start Thunderbird you
1326 will be alerted that it is not your default email application. (The default email application is the
1327 program that opens, for example, when you click a link on a web page to an email address.) If
1328 you want Thunderbird to be the default email application, click **Yes** to set it as your default
1329 mailer. If not (for example if you are just trying out Thunderbird) click **No**.

1330 2.11.5 Thunderbird Configuration for use with Microsoft Exchange

1331 Thunderbird can be used to access Microsoft Exchange servers that support IMAP or POP3. The
1332 normal way to use Thunderbird with a Microsoft Exchange Server requires the system
1333 administrator to enable the POP/IMAP/SMTP mail servers that are bundled with that server.
1334 Otherwise, since Exchange uses a proprietary MAPI protocol, accessing Exchange from
1335 Thunderbird can require a plugin or gateway¹ that provides standard, compliant protocols in
1336 front of proprietary Exchange (e.g., DavMail, ExQuilla).

1337 In setting up Thunderbird:

- 1338 1. Open Thunderbird and click the **Tools** menu option. Click **Account Settings**. Click **Account**
1339 **Settings** again to start the process for the Exchange connection.
- 1340 2. Enter the full name at the first window. This name is what email recipients see in their
1341 inbox. In the following text box, enter your email address. Click the **Next** button.
- 1342 3. Select **IMAP Mail Server** from the drop-down window. Enter the Exchange server name in
1343 the **IMAP Server Name** text box. In the **Outgoing Server** text box, enter the Exchange server
1344 name again. Click the **Next** button.
- 1345 4. Check the box labeled **Username** and **password**. Enter your current username used to log
1346 into the machine. Remove the check mark in the box labeled Use secure connection. Click
1347 **Finish**. The Thunderbird application is ready to send and receive email from the Exchange
1348 server.

1349 2.11.6 Thunderbird Configuration for use with Dovecot/Postfix

1350 General step-by-step instructions for setting up Thunderbird can be found at
1351 https://products.secureserver.net/email/email_thunderbird.htm (*Setting Up Your POP or IMAP*
1352 *Email Address with Mozilla Thunderbird*).

1353 Instructions for automatic account configuration can be found at
1354 <https://support.mozilla.org/en-US/kb/automatic-account-configuration>. Manual account
1355 configuration requires the following information:

- 1356 ■ incoming mail server and port (for example, **pop.example.com** and port 110 or
1357 **imap.example.com** and port 143)
- 1358 ■ outgoing mail server and port (for example, **smtp.example.com** and port 25)

1. Several links to free and commercial gateway and add-on products can be found by using a search engine with the argument “how to configure Microsoft Exchange server in Thunderbird.”

1359 ■ security setting for the connection with the server (for example, **STARTTLS** or **SSL/TLS** and
1360 whether or not to use secure authentication)

1361 Instructions can be found at

1362 <https://support.mozilla.org/en-US/kb/manual-account-configuration>.

1363 2.11.7 Thunderbird Support

1364 Although it is open source software, Thunderbird support is available from Mozilla and other
1365 sources.

3 Device Configuration and Operating Recommendations

3	3.1 Using SSL for Cryptographic Certificate Generation.....	54
4	3.2 Cryptographic Operations (User Actions).....	60
5	3.3 Server-to-Server Encryption Activation and Use.....	67
6	3.4 Utilities and Useful Tools	67
7		

This section provides additional information regarding for installing, configuring and operating Email and DNS security applications. [Section 3.1](#) provides specific recommendations regarding certificate generation. [Section 3.2](#) describes cryptographic operation and management by users on Outlook and Thunderbird. [Section 3.3](#) describes setting up Exchange and Postfix MTAs to provide server-to-server encryption of email. [Section 3.4](#) provides links to some tools and utilities that are useful in installing, configuring, provisioning, and maintaining DNS-based email security software.

It is recommended that the installation, configuration, and operation of DNS servers be conducted in conformance to NIST SP 800-81-2, the *Secure Domain Name System (DNS) Deployment Guide*. [Appendix D](#) provides a checklist for management of secure DNSs. Installation, configuration, and operation of email applications should follow the recommendations of SP 800-177, *Trustworthy Email*.

3.1 Using SSL for Cryptographic Certificate Generation

OpenSSL is a widely used open-source implementation of TLS/SSL and supporting cryptographic libraries for various version of Linux, but can also be used with Mac OS. OpenSSL also contains user utilities for generating cryptographic keys, certificate requests, and X.509 certificates. There is a FIPS-140 approved version of relevant OpenSSL cryptographic modules available for use by federal agencies.

3.1.1 OpenSSL Installation Basics and System Requirements

OpenSSL components and libraries are often standard components in base Linux installs, or can be installed using the built-in repository management system used with the version of Linux in use (e.g. apt-get, yum, rpm, etc.). Administrators may wish to install the developer repositories (*-devel or *-src) to make sure that all necessary header files are installed to support server implementations that rely on OpenSSL for cryptographic support. The latest version of OpenSSL, as well as FIPS approved versions may not be available in repositories and may need to be built from source from the OpenSSL project homepage¹.

In addition to having a base supported operating system, OpenSSL requires Perl 5 and a C compiler and development environment (with tools like **make**) to be successfully compiled and installed.

3.1.1.1 OpenSSL FIPS Approved Installation

Federal agencies or other organizations that are required to use FIPS-140 approved cryptographic modules can use OpenSSL FIPS approved version. These necessary modules are not always available via OS-specific repositories, but must be manually downloaded and compiled. The newly compiled libraries then replace any older, or pre-installed versions². Server daemons (e.g. BIND named, postfix, etc.) that rely on OpenSSL for cryptographic support will then use the FIPS-140 approved version of the libraries.

1. <https://openssl.org/>

2. https://wiki.openssl.org/index.php/Compilation_and_Installation#FIPS_Capable_Library

44 3.1.1.2 OpenSSL Installation on Mac OS

45 Normally, there is no need to install a separate set of cryptographic libraries for Mac OS.
 46 OpenSSL is installed in the standard Mac OS distribution provides the same functionality
 47 However, if there is a desire to upgrade the standard installation an alternative repository tool
 48 (e.g. homebrew¹) may be necessary or certain files need to be changed² in order to build
 49 OpenSSL on an Apple system.

50 3.1.2 OpenSSL Configuration

51 3.1.2.1 Configuration of OpenSSL to act as a Local Certificate Authority (CA)

52 OpenSSL can be used to generate certificates and act as a local enterprise Certificate Authority
 53 (CA). This is not always advisable as it is very bare-boned set of tools. Enterprises using OpenSSL
 54 as their CA must take great care to insure that the root certificate (i.e. the CA certificate that
 55 signs all the end-entity certificates) is adequately protected. Compromise of the root certificate
 56 private key would allow an attacker to generate arbitrary certificates for spoofed hosts and
 57 services. How this root certificate private key is protected is beyond the scope of this document
 58 but should include adequate physical, access, and logical controls.

59 OpenSSL can be used via the `openssl` command line tool to generate key pairs, and certificates
 60 for those key pairs. This certificate generation can be done by adding the certificate data on the
 61 command line, or using a configuration file for (organizational) default values. For example, if
 62 the organizational policy is for all certificates to have a lifetime of one year (365 days), that
 63 value can be set in a configuration file and does not need to be set using command line options
 64 unless there is a need to override the default for a specially generated certificate.

65 The general order in setting up OpenSSL to operate an enterprise local CA (or to generate
 66 self-signed certificates) is to: Generate and set up configuration files, generate the root
 67 certificate, and finally, generate and sign end entity certificates.

68 3.1.2.2 The OpenSSL CA Configuration File

69 Once OpenSSL is installed on the system, the CA admin needs to find and edit the `openssl.cnf`
 70 configuration file. Where this file is located depends on how OpenSSL was installed on the
 71 system. Many repository installations will put the file at `/etc/ssl/openssl.cnf` but it may
 72 also be found at `/usr/ssl` or `/usr/openssl` or some other directory.

73 The configuration file is broken down into blocks around `openssl` commands. Most of these
 74 blocks can be left in their default values unless there is a specific policy reason for changing
 75 them. The two blocks that enterprise CA admins will likely need to change is `[CA_default]` and
 76 `[req]` which contain the default values for cryptographic and hash algorithms, default sizes and
 77 lifetimes, and Distinguished Name (country, organizational name, Common Name, etc.)
 78 respectively. An example snippet of the configuration file `openssl.cnf` is given in [figure 3.1](#)
 79 below.

1. <http://brew.sh/>

2. https://wiki.openssl.org/index.php/Compilation_and_Installation#Mac

80 The values in the [**CA_defaults**] block deal with the components of the CA itself: the
81 directories used, the serial number file, etc. These are used to manage the CA itself, not directly
82 involved with the cryptographic operation of generating key pairs and certificates. CA
83 administrators can set these values to the appropriate directories for their enterprise CA.
84 OpenSSL does not generate some of the necessary directories and files (such as **serial**, which
85 keeps track of the serial numbers of issued certificates). These will need to be created by the
86 admin using a text editor or standard Linux commands.

87 The values in the [**req**] block deal with the identification data and characteristics of X.509
88 certificates generated by the CA. These values will most likely need to be edited by enterprise
89 CA administrators. If the enterprise certificate policy dictates that some values must be
90 constant across the organization, it makes sense to make them the default values in the
91 configuration file. For example, the enterprise always wants its HQ location used as the country,
92 state, and locality in every certificate it generates.

93

Figure 3.1 Example OpenSSL Configuration File

```

[ CA_default ]

dir      = /etc/pki/CA      # Where everything is kept
certs   = $dir/certs        # Where the issued certs are kept
crl_dir  = $dir/crl         # Where the issued crl are kept
database = $dir/index.txt  # database index file.
#unique_subject = no        # Set to 'no' to allow creation of
                           # several certificates with same subject.
new_certs_dir = $dir/newcerts # default place for new certs.

certificate = $dir/cacert.pem      # The CA certificate
serial     = $dir/serial          # The current serial number
crlnumber  = $dir/crlnumber      # the current crl number
                           # must be commented out to leave a V1 CRL
crl      = $dir/crl.pem         # The current CRL
private_key = $dir/private/cakey.pem# The private key
RANDFILE   = $dir/private/.rand   # private random number file

x509_extensions = usr_cert      # The extensions to add to the cert

[ req ]
default_bits = 2048
default_md   = sha256
default_keyfile = privkey.pem
distinguished_name = req_distinguished_name
attributes    = req_attributes
x509_extensions = v3_ca

[ req_distinguished_name ]
countryName          = Country Name (2 letter code)
countryName_default  = XX
countryName_min      = 2
countryName_max      = 2

stateOrProvinceName  = State or Province Name (full name)
#stateOrProvinceName_default = Default Province

localityName          = Locality Name (eg, city)
localityName_default  = Default City

0.organizationName    = Organization Name (eg, company)
0.organizationName_default = Default Company Ltd

organizationalUnitName = Organizational Unit Name (eg, section)
#organizationalUnitName_default = 

commonName          = Common Name (eg, your name or your server's hostname)
commonName_max       = 64

emailAddress         = Email Address
emailAddress_max     = 64

```

94

95

The enterprise CA admin can then put these entries in the appropriate line in the configuration file. For example:

```

97 [ req_distinguished_name ]
98 countryName          = Country Name (2 letter code)
99 countryName_default  = US
100 countryName_min     = 2

```

```

101    countryName_max          = 2
102
103    stateOrProvinceName      = State or Province Name (full name)
104    stateOrProvinceName_default = District of Columbia
105
106    localityName             = Locality Name (eg, city)
107    localityName_default      = Washington
108
109    0.organizationName        = Organization Name (eg, company)
110    0.organizationName_default = Department of Examples

```

111 Once the default values are in place, the configuration file will be used unless overridden in the
 112 `openssl` command line. If the configuration file has been moved to a new directory, the
 113 command line option `-config` should be included in the `openssl` command to point to the
 114 location of the new configuration file location.

115 3.1.2.3 Using Linux Environment Variables to Dynamically Set Common Name and 116 SubjectAltName

117 Not all of the values can be set via the command line override. The most important value that
 118 an enterprise CA admin may want to change is the **subjectAltName** of a certificate. The
 119 **subjectAltName** is used to provide alternative hostnames for a server that can be checked
 120 during PKIX validation. This allows one server to have multiple names and still use the same key
 121 pair for TLS. The **subjectAltName** default can be set in the configuration file, but cannot be set
 122 at the command line.

123 On Linux systems, the following can be used in the configuration file to use environment
 124 variables for CommonName (called **COMNAME**) and SubjectAltName (called **SAN**). See below:

```

125    commonName                = Common Name (eg, your name or your
126    commonName_default         = ${ENV::COMNAME}
127    commonName_max             = 64
128
129    subjectAltName             = ${ENV::SAN}
130

```

131 After the changes have been made to the configuration file, the **CommonName** and
 132 **SubjectAltName** can be set dynamically (either via command line or appropriate system call in
 133 scripts, programs, etc.) to set the entries before generating a certificate.

134 3.1.3 Certificate Generation

135 3.1.3.1 Generate the Root Certificate

136 Once the configuration file is edited, the enterprise CA administrator must first generate a root
 137 certificate. This can be done using the `openssl` command line tool, or an included support script
 138 `CA.pl`. The following examples use the command line, as it is flexible and can be used via
 139 scripted system calls (that set environment variables, etc.). The basic command to generate a
 140 root certificate is:

```

141 >openssl req -config <config file> \
142   -key private/ca.key.pem \
143   -new -x509 -days 7300 -sha256 -extensions v3_ca \
144   -out certs/ca.cert.pem

```

145 Here the **-config** option is used to list the location of the configuration file in use. The use of the
146 **-days** option is to increase the lifetime of the root cert over any default value in the
147 configuration file. The root certificate is not like end-entity issued certificates and often
148 requires more configuration possible manual installation in enterprise systems, so should be
149 longer lived for administration purposes (and highly protected). Enterprise CA administrators
150 should consult NIST SP 800-152, *A Profile for U. S. Federal Cryptographic Key Management*
151 *Systems* for recommendations on how to set up a key management system.

152 3.1.3.2 Generating Intermediate and End-Entity Certificates

153 Once the CA infrastructure is set up and the root certificate is generated, the enterprise CA can
154 start generating end-entity and (if desired) intermediate certificates. Intermediate certificates
155 are just that: certificates that are extra “links” in the PKIX validation chain to the root certificate.
156 They are not usually installed as trust anchors, but can be used to sign other (often end-entity)
157 certificates.

158 The advantage of using intermediate certificates is that they can be used to compartmentalize
159 end-entity certificates, so a compromise of an intermediate cert means that only that
160 certificate (and those it signed) are compromised, and not the entire CA. Intermediate
161 certificates also allow CA administrators to keep the root certificate safely stored offline. Once
162 the root key is used to sign the intermediate certificates, it can be stored offline until new
163 intermediate certificates are needed.

164 The disadvantages of using intermediate certificates is that they are needed by all clients
165 wishing to do PKIX validation. If a client cannot find (or have stored) all necessary intermediate
166 certificates, it cannot validate all end-entity certificates. Protocols like TLS account for this by
167 having certificate chains available (end-entity and necessary intermediate certificates), but not
168 all protocols do this. DANE is an option for publishing intermediate certificates in the DNS as
169 intermediate certs, or as short-circuited trust anchors, depending on which Certificate Usage
170 (CU) parameter is used [RFC6698].

171 The general command to generate a new client key pair and certificate is:

```

172 >openssl req -new -nodes -config <config file> -keyout <key filename>
173   -out \
174     <CSR filename>

```

175 The above command will generate a key pair and a Certificate Signing Request (CSR) for the
176 new certificate. The **-nodes** option disables the setting of a password for decrypting the private
177 portion of the key pair. This is important to set for server certificates where there is no end user
178 to enter a password (and the private key is needed to set up a TLS connection). For
179 intermediate certificates, this should not be set, as that private key should be protected.

180 Once the CSR is generated, it is made into a certificate:

```

181 >openssl ca -config <config file> -out -infiles <CSR filename> -out
182   <cert name>

```

183 Then the administrator follows the prompt. Administrators using intermediate keys may also
 184 use the **-key <private key>** option to have openssl use the desired intermediate key.
 185 Alternatively, the administrator could configure the which signing key to use in the openssl
 186 configuration file. Indeed, several separate configuration files could be used if multiple
 187 intermediate keys are used for the enterprise CA.

188 Once the new certificate and key pair have been generated, they must be protected from
 189 unauthorized disclosure. They must be securely communicated to server administrators so the
 190 administrators can configure them for use. Once the key has outlived its lifetime, it must be
 191 securely retired and removed. These operations should be documented as part of the
 192 enterprise key management system.

193 3.2 Cryptographic Operations (User Actions)

194 This section provides information regarding user actions necessary for users to invoke digital
 195 signature, encryption, and cryptographic certificate management features of Outlook and
 196 Thunderbird. The user's experience varies from relatively minimal additional impact in
 197 enterprise environments with established system administration and support to a significant
 198 impact in the case of individual self-supported users. Where the enterprise offers systems
 199 administration and support services, the user's experience with respect to DNS services is
 200 essentially unchanged. One exception is that, where DNS authentication fails, email messages
 201 sent to or by a user will not be delivered. This should be an uncommon experience for
 202 correspondents but it is up to the enterprise DNS administrator to prevent this happening.
 203 Similarly, for server-to-server encryption, the security protection features should be essentially
 204 transparent to the user.

205 3.2.1 Outlook

206 To use digital signatures and encryption, both the sender and recipient must have a mail
 207 application that supports the S/MIME standard. Outlook supports the S/MIME standard.

208 Instructions for user-driven cryptographic functions vary from version to version and platform
 209 to platform. Accessing **digital signature** on an Outlook **Help** page usually provides the necessary
 210 operator instructions. The example instructions provided here are for Outlook 2016 for
 211 Windows 10 and Outlook for Mac 2011.

212 3.2.1.1 Outlook 2016 for Windows 10

213 When a user has been issued an S/MIME certificate they can import it into the Outlook 2016's
 214 Trust Center to be used for digital signature and encryption based upon the key usages of the
 215 certificate. When a smart card containing a secure email digital signature certificate is inserted
 216 the Windows operating system, the OS will import the certificate into the user's personal
 217 certificate store. This will occur when the user inspects the smart card with the **certutil.exe**
 218 **-scinfo** command or if the following group policy is enabled:

219 **Computer Configuration -> Administrative Templates -> Windows Components -> Smart Card:**
 220 **Turn on certificate propagation from smart card**

221 To view the certificates in the user's certificate store, type **certmgr.msc**.

222 Configure Outlook 2106 S/MIME Settings:

223 1. Open Outlook 2016.

224 2. Click on **File**, and then **Options**.

225 3. In the left-hand menu click on **Trust Center**.

226 4. Click on the **Trust Center Settings** box.

227 5. Click **Email Security** in the left-hand menu.

228 6. Click the **Settings** button within the Encrypted Email section.

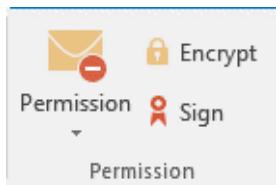
229 7. Enter a name within the **Security Settings Name** field.

230 8. Select the Signing Certificate by clicking on the **Choose** button for the signing certificate and select the **Hash Algorithm**.

231 9. If you have an S/MIME encryption certificate select the **Choose** button for the encryption certificate and select the **Encryption Algorithm**.

232 10. Select the radio button **Send** to send these certificates with signed messages.

233 The user can choose to always digitally sign a message by selecting the **Add digital signature to outgoing messages** within the **Trust Center -> Email Security -> Encrypted Email** menu. This will digitally sign every outgoing email. To individually sign an email, within the draft message itself go to **Options** and within the **Permissions** menu select the **Sign** icon.



239

240 3.2.1.2 Outlook for Mac 2011 Certificate Management

241 If the user has a person's certificate in Outlook, he or she can validate a digitally signed
242 message.¹

243 1. Importing a Certificate

244 a. At the bottom of the navigation pane, click **Contacts** .

245 b. Open the desired contact, and then click the **Certificates** tab.

246 c. Click , locate the certificate, and then click **Open**.

247 Note: To set the default certificate for a contact, select the certificate, click , and
248 then click **Set as Default**.

249 2. Exporting a Certificate

250 Certificates can be exported in three formats: DER encoded X.509, PEM (Base-64 encoded
251 X.509), and PKCS #7. The DER encoded X.509 format is the most common, but the user
252 might want to ask what format his or her recipient requires.

1. This also enables the user to send that person an encrypted message (user to user).

253 a. At the bottom of the navigation pane, click **Contacts** .

254 b. Open the desired contact, and then click the **Certificates** tab.

255 c. Select the certificate, click , and then click **Export**. To set the format of the certificate, make a selection on the **Format** menu.

256 3. Deleting a Certificate

258 a. At the bottom of the navigation pane, click **Contacts** .

259 b. Open the desired contact, and then click the **Certificates** tab.

260 c. Select the certificate, and then click .

261 3.2.1.3 Digital Signature

262 To use digital signatures (or encryption), both the sender and recipient must have a mail
263 application that supports the S/MIME standard. Outlook supports the S/MIME standard.

264 Note: *Before a user starts this procedure, he or she must first have a certificate added to the*
265 *keychain on his or her computer. For information about how to request a digital certificate from a*
266 *certification authority, see Mac Help.*

267 1. On the **Tools** menu, click **Accounts**.

268 The user clicks the account from which he or she wants to send a digitally signed message,
269 clicks **Advanced**, and then clicks the **Security** tab.

270 2. Under **Digital signing**, on the **Certificate** pop-up menu, the user clicks the certificate that he
271 or she wants to use.

272 Note: The **Certificate** pop-up menu only displays certificates that are valid for digital
273 (signing or encryption) that the user has already added to the keychain for his or her Mac
274 OS X user account. To learn more about how to add certificates to a keychain, see Mac OS
275 Help.

276 3. To make sure that the user's digitally signed messages can be opened by all recipients, even
277 if they do not have an S/MIME mail application and cannot verify the certificate, select the
278 **Send digitally signed messages as clear text** check box.

279 4. Click **OK**, and then close the **Accounts** dialog box.

280 5. In an e-mail message, on the **Options** tab, click **Security**, and then click **Digitally Sign**
281 **Message**.



283 6. Finish composing the message, and then click **Send**.

284 3.2.2 Thunderbird¹

285 For purposes of illustration, the description of the user experience with Thunderbird also
 286 included certificate management requirements. The example here shows both S/MIME and
 287 PGP examples of certificate management. The S/MIME approach is recommended. Note that
 288 when using OpenPGP, a FIPS 140-conformant version should always be used.

289 3.2.2.1 S/MIME Certificate Management

290 S/MIME certificates are used for digitally signed and encrypted e-mail messages. For
 291 information about getting or creating S/MIME certificates, see:
 292 http://kb.mozilla.org/Getting_an_SMIME_certificate.

293 1. Installing an S/MIME certificate

294 Important: Before a user can create or import his or her own certificate and private key, he
 295 or she must first set a master password if this has not already been done. The master
 296 password is needed so that imported certificates are stored securely. See
 297 http://kb.mozilla.org/Master_password for instructions for setting a master password.

298 The user may have his or her own personal certificate and private key in a .p12 or .pfx file,
 299 and may wish to import it into Thunderbird. Once a Master Password has been set, the user
 300 can import/install a personal S/MIME certificate from a .p12 or .pfx file by doing the
 301 following steps.

- 302 a. Open the Certificate Manager by going to **Tools -> Options... -> Advanced ->**
Certificates -> Manage Certificates....
- 303 b. Go to the tab named **Your Certificates**.
- 304 c. Click on **Import**.
- 305 d. Select the **PCKS12** certificate file (.pfx or .p12).
- 306 e. It will ask the user for the master password for the software security device. The user
 307 enters his or her master password and clicks **OK**.
- 308 f. Next, it will ask the user for the password protecting his or her personal certificate. If
 309 the user's .p12 or .pfx file has a password, the user enters it here, otherwise leave this
 310 field empty. The user then clicks **OK**.

312 The S/MIME certificate should now have been imported. If the certificate was not
 313 trusted, consult the instructions at
 314 http://kb.mozilla.org/Thunderbird_-_FAQs_-_Import_CA_Certificate.

315 2. Configuring Thunderbird for using the certificate to sign email

316 Go to **Tools -> Account Settings...** in Thunderbird. Then find the account with the email
 317 address that matches the email address in the certificate that has just been installed. The
 318 user chooses **Security** under that account and selects the certificate that has just been
 319 installed. The rest of the options should be self explanatory. When the user selects a
 320 certificate in Account Settings, that selection only applies to the account's default identity
 321 or identities. There is no user interface for specifying certificates for an account's other

1. See <https://support.mozilla.org/en-US/kb/digitally-signing-and-encrypting-messages>

322 identities. If desired, this can be worked around by editing the settings manually, copying
 323 the settings from an account's default identity to some other identity. The settings have
 324 names ending in: **signing_cert_name**, **sign_mail**, **encryption_cert_name**, and
 325 **encryptionpolicy**.

326 3. User Installation of a Self-Signed S/MIME Certificate

327 If the SMIME certificate in a user's .p12 or .pfx file is a self-signed certificate for the user's
 328 own identity, then before that file can be installed into the tab named **Your Certificates**, the
 329 user must first install that certificate as a certificate authority in the **Authorities** tab. The
 330 PKCS12 certificate file will not install into the **Authorities** tab. The user will need a copy of a
 331 self-signed certificate that does not contain the user's private key. This is usually in the form
 332 of a .cer file. One way to obtain the .cer form of a certificate from the .p12 file is to use the
 333 Firefox Add-on Key Manager to extract the .cer certificate from the .p12 file. With that
 334 Add-on installed in Thunderbird, the user goes to **Tools -> Key Manager Toolbox -> Key**
 335 **Manager -> Your Keys**, selects his or her key, selects **Export** and chooses **X.509** as file
 336 format.

- 337 a. Go to **Tools -> Options... -> Advanced -> Certificates -> Manage Certificates....**
- 338 b. Go to the **Authorities** tab.
- 339 c. Click on **Import**.
- 340 d. Select the **.cer** file.
- 341 e. It will ask the user for what purposes he or she wants to trust the certificate. The user
 342 selects **Trust this CA to identify email users**.
- 343 f. Click **OK** to complete the import.

344 Note: *Thunderbird automatically adds other people's S/MIME certificates to the **Other People's***
 345 *tab of a user's Certificate Manager when he or she receives from them a digitally signed*
 346 *message with a valid signature and with an S/MIME certificate issued by a recognized and*
 347 *trusted Certificate Authority (CA). CA certificates that appear in Thunderbird's **Authorities** tab*
 348 *are recognized, and may also be trusted. CA certificates that do not appear in that tab are*
 349 *considered **unrecognized**. An S/MIME certificate that was issued by an unrecognized CA will*
 350 *not be automatically added to the **Other People's** tab of the user's Certificate Manager. If the*
 351 *user attempts to manually import an S/MIME certificate that was issued by an unrecognized CA,*
 352 *nothing will happen--literally. Thunderbird will not even display an error dialog. It will just not*
 353 *import the S/MIME certificate. This is generally not a problem when receiving an S/MIME*
 354 *certificate that was issued by a trusted Certificate Authority (CA), but could be a problem for a*
 355 *certificate that was issued by an unrecognized or untrusted CA, or for a certificate that is*
 356 *self-signed (i.e. it has no CA other than itself). So, before a user can import an S/MIME*
 357 *certificate that is issued by an unrecognized CA or is self-signed, he or she must first acquire*
 358 *and import the certificate for the issuing CA. In the case of a self-signed certificate, a **.cer** file*
 359 *needs to be acquired from the individual whose certificate the user wishes to add.*

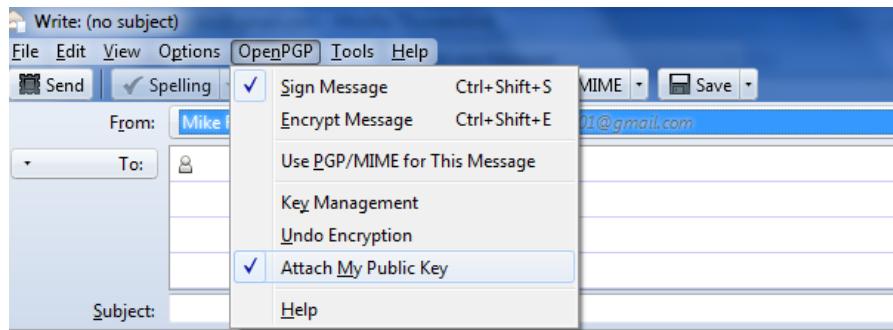
360 3.2.2.2 PGP Example of Sending and Receiving Public Keys

361 1. Sending a public key via email

362 To send signed messages to other people that the recipients can validate, the user must first
 363 send them the public key:

- 364 a. Compose the message.

365 b. Select **OpenPGP** from the Thunderbird menu bar and select **Attach My Public Key**.



366

367 c. Send the email as usual.

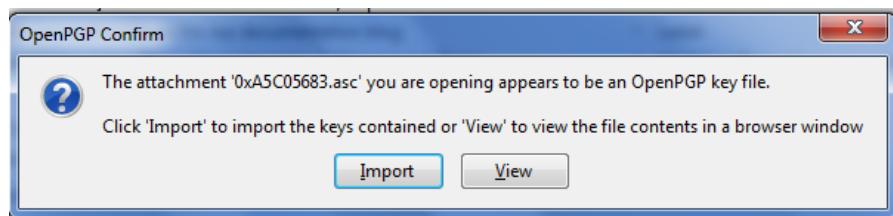
368 2. Receiving a public key via email

369 To verify signed messages from other people, the public key must be received and stored:

370 a. Open the message that contains the public key.

371 b. At the bottom of the window, double click on the attachment that ends in **.asc**. (This file
372 contains the public key.)

373 c. Thunderbird automatically recognizes that this is a PGP key. A dialog box appears,
374 prompting the **Import** or **View** of the key. Click **Import** to import the key.



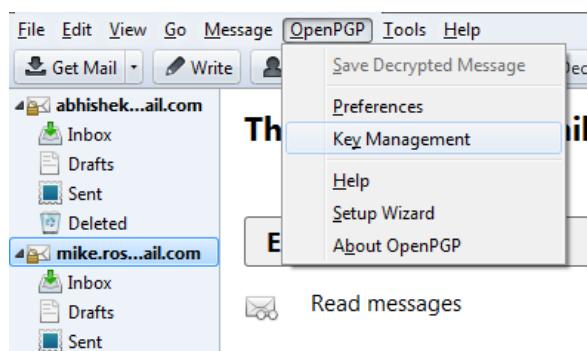
375

376 d. A confirmation that the key has been successfully imported will be shown. Click **OK** to
377 complete the process.

378 3. Revoking a key

379 If the private key may have been “compromised” (that is, someone else has had access to
380 the file that contains the private key), revoke the current set of keys as soon as possible and
381 create a new pair. To revoke the current set of keys:

382 a. On the Thunderbird menu, click **OpenPGP** and select **Key Management**.



383

384 b. A dialog box appears as shown below. Check **Display All Keys by Default** to show all the
 385 keys.

386 c. Right-click on the key to be revoked and select **Revoke Key**.

387 d. A dialog box appears asking the user if he or she really want to revoke the key. Click
 388 **Revoke Key** to proceed.

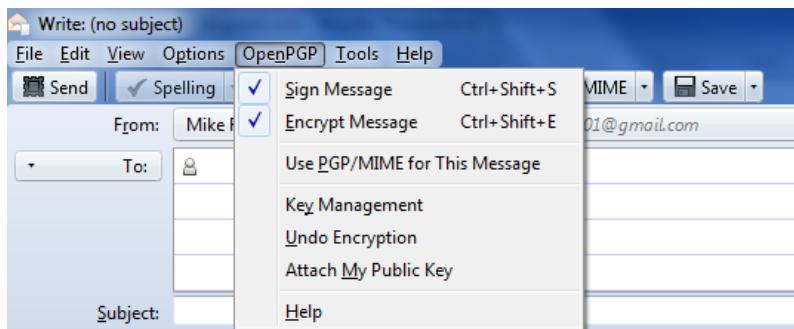
389 e. Another dialog box appears asking for the entry of a secret passphrase. Enter the
 390 passphrase and click **OK** to revoke the key.

391 The user sends the revocation certificate to the people with whom he or she corresponds so
 392 that they know that the user's current key is no longer valid. This ensures that if someone tries
 393 to use the current key to impersonate the user, the recipients will know that the key pair is not
 394 valid.

395 3.2.2.3 Sending a Digitally Signed Email

396 1. Compose the message as usual.

397 2. To digitally sign a message, select **OpenPGP** from the Thunderbird menu and enable the
 398 **Sign Message** option.

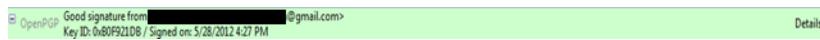


399 3. If the email address is associated with a cryptographic certificate, the message will be
 400 signed with the key contained in that certificate. If the email address is not associated with
 401 a cryptographic certificate, a certificate must be selected from a list.
 402 4. Send the message as usual.

404 3.2.2.4 Reading a Digitally Signed Email

405 When a signed message is received, and If Thunderbird recognizes the signature, a green bar
 406 (as shown below) appears above the message.

407 To determine whether or not the incoming message has been signed, look at the information
 408 bar above the message body.¹



410 If the message has been signed, the green bar also displays the text, "Signed message".

1. If the message is also encrypted on a user to user basis, Thunderbird will also ask for the entry of a secret passphrase to decrypt the message.

411 A message that has not been signed could be from someone trying to impersonate someone
 412 else.

413 3.3 Server-to-Server Encryption Activation and Use

414 3.3.1 Office 365 Exchange

415 Server-to-server encryption (Scenario 1) is available on Exchange for Office 365. Office 365
 416 encrypts users' data while it's on Microsoft servers and while it's being transmitted between
 417 the user and Microsoft. Office 365 provides controls for end users and administrators to fine
 418 tune what kind of encryption is desired to protect files and email communications. Some
 419 technical library links for specific topics are as follows:

- 420 ■ Information on encryption using Office 365 Exchange can be found at
 421 <https://technet.microsoft.com/en-us/library/dn569286.aspx>.
- 422 ■ Information regarding the different types of email encryption options in Office 365
 423 including Office Message Encryption (OME), S/MIME, Information Rights Management
 424 (IRM) can be found at <https://technet.microsoft.com/en-us/library/dn948533.aspx>.
- 425 ■ Information regarding definition of rules regarding email message encryption and
 426 decryption can be found at <https://technet.microsoft.com/en-us/library/dn569289.aspx>.
- 427 ■ Information regarding sending, viewing, and replying to encrypted messages can be found
 428 at <https://technet.microsoft.com/en-us/library/dn569287.aspx>.
- 429 ■ Service information for message encryption can be found at
 430 <https://technet.microsoft.com/en-us/library/dn569286.aspx>.

431 3.3.2 Postfix

432 Postfix TLS support is described at http://www.postfix.org/TLS_README.html. Postfix can be
 433 set to encrypt all traffic when talking to other mail servers.¹

434 3.4 Utilities and Useful Tools

435 This section provides links to some tools and utilities that are useful in installing, configuring,
 436 provisioning, and maintaining DNS-based email security software.

1. "Setting Postfix to encrypt all traffic when talking to other mailservers," *Snapdragon Tech Blog*, August 9, 2013. <http://blog.snapdragon.cc/2013/07/07/setting-postfix-to-encrypt-all-traffic-when-talking-to-other-mailservers/>

437 3.4.1 DANE Tools

438 3.4.1.1 SMIMEA Retriever Tool

439 The SMIMEA retriever tool, developed by Santos Jha as part of this project, retrieves SMIMEA
440 records from a DNS for a given email address and stores the certificates in PKCS12 format. This
441 PKCS12 store can subsequently be imported into an MUA such as Thunderbird or Outlook. Since
442 this software is used for offline provisioning of certificates, the developer focused on selector=0
443 and matching type=0. It is written using Java 8.

444 3.4.1.2 TLSA Generator

445 Shumon Huque's online TLSA generator generates TLSA resource records from a certificate and
446 parameters for which prompts are included. The link to the tool is
447 https://www.huque.com/bin/gen_tlsa.

448 3.4.1.3 High Assurance Domain Toolbox

449 NIST's High Assurance Domain Toolbox is a collection of perl scripts used to generate and
450 format SMIMEA and TLSA RR's for use with the High Assurance Testbed. Each of these scripts
451 are used independently and not all required to be used if other solutions work better. The tool
452 can be found at <https://github.com/scottr-nist/HAD-tlsa-toolbox>.

453 3.4.1.4 Swede

454 Swede is a tool for use in creating and verifying DANE records. The tool can be found at
455 <https://github.com/pieterlexis/swede>.

456 3.4.1.5 Hash-slinger

457 Hash-slinger is a package of tools created by Paul Wouters of RedHat to make it easy to create
458 records for the DANE protocol that will allow you to secure your SSL/TLS certificates using
459 DNSSEC. The package is available for Linux at:
460 <http://people.redhat.com/pwouters/hash-slinger/>.

461 3.4.2 DANE Validation Sites and Testers

462 3.4.2.1 NIST DANE Testers

463 NIST's DANE-testers for RFC 6698 conformance can be found at
464 <http://dane-test.had.dnsops.gov/>.

465 3.4.2.2 SMIMEA Test Tool

466 Grier Forensics' SMIMEA Test tool can be found at <http://dst.grierforensics.com/#/start>.

467 **3.4.2.3 DANE Validator Online Test Tool**468 The DANE validator online test tool found at <https://check.sidnlabs.nl/dane/> attempts to
469 perform validation of a TLSA/PKI pair according to the DANE Internet standard. Note that the
470 tool automatically selects Port 443 and TCP. SNI support is included. The tool set uses the
471 Idns-dane example from LDNS from NLnet Labs.472 **3.4.2.4 DANE SMTP Validator**473 The DANE SMTP Validator, an SMTP DANE test tool, can be found at <https://dane.sys4.de/>.474 **3.4.3 Other Test Tools**475 DNSViz is a tool for visualizing the status of a DNS zone. It was designed as a resource for
476 understanding and troubleshooting deployment of the DNS Security Extensions (DNSSEC). It
477 provides a visual analysis of the DNSSEC authentication chain for a domain name and its
478 resolution path in the DNS namespace, and it lists configuration errors detected by the tool.
479 This DNSSEC test tool is not DANE specific, but helpful. It can be found at <http://dansviz.net/>.

Appendix A Acronyms

2	ASN	Abstract Syntax Notation
3	AXFR	DNS Full Zone Transfer Query Type
4	BIND	Berkeley Internet Name Daemon
5	BSD	Berkeley Software Distribution
6	CA	Certificate Authority
7	CRL	Certificate Revocation List
8	CSR	Certificate Signing Request
9	CU	Certificate Usage Type
10	DANE	DNS-based Authentication of Named Entities
11	DNS	Domain Name System
12	DNSSEC	DNS Security Extensions
13	Email	Electronic Mail
14	FIPS	Federal Information Processing Standard
15	GAL	Global Address List
16	HTTP	Hypertext Transfer Protocol
17	IETF	Internet Engineering Task Force
18	IMAP	Internet Message Access Protocol
19	IP	Internet Protocol
20	ITL	Information Technology Laboratory
21	LDAP	Lightweight Directory Access Protocol
22	MIME	Multipurpose Internet Mail Extension
23	MTA	Mail Transfer Agent
24	MUA	Mail User Agent
25	MX	Mail Exchange (Resource Record)
26	NCCoE	National Cybersecurity Center of Excellence
27	NIST	National Institute of Standards and Technology
28	NSD	Network Server Daemon
29	OS	Operating System
30	PKI	Public Key Infrastructure
31	PKIX	Public Key Infrastructure X.509
32	POP	Post Office Protocol
33	RFC	Request for Comments

34	RMF	Risk Management Framework
35	RR	Resource Record
36	S/MIME	Secure/Multipurpose Internet Mail Extensions
37	SMIMEA	S/MIME Certificate Association (Resource Record)
38	SMTP	Simple Mail Transfer Protocol
39	SP	Special Publication
40	SQL	Structured Query Language
41	TLS	Transport Layer Security
42	TLSA	TLS Certificate Association (Resource Record)
43	UA	User Agent
44	VLAN	Virtual Local Area Network
45	VM	Virtual Machine

Appendix B References

2 *Security Requirements for Cryptographic Modules*, Federal Information Processing Standard
3 (FIPS), FIPS 140-2, May 2001 (including change notices as of 12-03-2002). <http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf>

4 *Guidelines on Electronic Mail Security*; NIST Special Publication; SP 800-45 Ver. 2; Tracy, Jansen,
5 Scarfone, Butterfield; February 2007. <http://csrc.nist.gov/publications/nistpubs/800-45-version2/SP800-45v2.pdf>

6 *Federal S/MIME V3 Client Profile*, NIST Special Publication, SP 800-49, Chernick, November
7 2002. <http://csrc.nist.gov/publications/nistpubs/800-49/sp800-49.pdf>

8 *Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS)
9 Implementations*; NIST Special Publication; SP 800-52 Rev. 1; Polk, McKay, Chokhani;
10 April 2014. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-52r1.pdf>

11 *Security and Privacy Controls For Federal Information Systems And Organizations*, NIST Special
12 Publication, SP 800-53 Rev. 4, Joint Task Force Transformation Initiative, April 2013.
13 <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf>

14 *Recommendation for Key Management: Part 1 - General*, NIST Special Publication 800-57 Rev.4,
15 Barker, January 2016. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf>

16 *Electronic Authentication Guideline*; SP 800-63-2; Burr, Dodson, Newton, Perlner, Polk, Gupta,
17 Nabbus; August 2013. doi:10.6028/NIST.SP.800-63-2

18 *Secure Domain Name System (DNS) Deployment Guide*, NIST Special Publication, SP 800-81-2,
19 Chandramouli and Rose, September 2013. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-81-2.pdf>

20 *A Framework for Designing Cryptographic Key Management Systems*; NIST Special Publication;
21 SP 800-130; Barker, Branstad, Smid, Chokhani; August 2013. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-130.pdf>

22 *A Profile for U.S. Federal Cryptographic Key Management Systems (CKMS)*; Third Draft; NIST
23 Special Publication; SP 800-152; Barker, Smid, Branstad; December 18, 2014. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-152.pdf>

24 *Trustworthy Email*; NIST Special Publication; SP 800-177; Chandramouli, Garfinkle, Nightingale
25 and Rose; Draft Publication; September 2016. <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-177.pdf>

26 *X.509 Certificate Policy for the U.S. Federal PKI Common Policy Framework*, Version 1.21. <http://www.idmanagement.gov/documents/common-policy-framework-certificate-policy>

27 *Domain Names - Concepts And Facilities*, RFC 1034, Mockapetris, November 1987. <https://www.ietf.org/rfc/rfc1034.txt>

28 *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*; IETF RFC 2459; Housley,
29 Ford, Polk, Solo; January 1999. <https://www.rfc-editor.org/rfc/rfc2459.txt>

30 *S/MIME Version 3 Message Specification*, IETF RFC 2633, Ramsdell, Ed., June 1999. <https://www.ietf.org/rfc/rfc2633.txt>

41 *Secret Key Transaction Authentication for DNS (TSIG)*; RFC 2845; Vixie, Gudmundsson, Eastlake,
42 and Wellington; May 2000. <https://www.ietf.org/rfc/rfc2845.txt>

43 *Secure Domain Name System (DNS) Dynamic Update*, RFC 3007, Wellington, November 2000.
44 <https://www.ietf.org/rfc/rfc3007.txt>

45 *ISO/IEC 9798-3 Authentication SASL Mechanism*, RFC 3163, Zuccherato and Nystrom, August
46 2001. <https://tools.ietf.org/html/rfc3163>

47 *SMTP Service Extension - Secure SMTP over TLS*, RFC 3207, Hoffman, February 2002. <https://www.ietf.org/rfc/rfc3207.txt>

49 *Cryptographic Message Syntax (CMS)*, RFC 3369, Housley, August 2002. <https://www.ietf.org/rfc/rfc3369.txt>

51 *Cryptographic Message Syntax (CMS) Algorithms*, RFC 3370, Housley, August 2002. <https://tools.ietf.org/html/rfc3370>

53 *Threat Analysis of the Domain Name System (DNS)*, IETF RFC 3833, Atkins and Austein, August
54 2004. <https://tools.ietf.org/html/rfc3833>

55 *Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1*, Certificate Handling RFC
56 3850, Ramsdell, July 2004. <https://tools.ietf.org/html/rfc3850>

57 *Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1*, Message Specification,
58 RFC 3851, Ramsdell, July 2004. <https://www.ietf.org/rfc/rfc3851.txt>

59 *DNS Security Introduction and Requirements*; RFC 4033; Arends, Austein, Larson, Massey, and
60 Rose; March 2005. <https://www.ietf.org/rfc/rfc4033.txt>

61 *Resource Records for the DNS Security Extensions*; RFC 4033; Arends, Austein, Larson, Massey,
62 and Rose; March 2005. <https://www.ietf.org/rfc/rfc4034.txt>

63 *Protocol Modifications for the DNS Security Extensions*; RFC 4033; Arends, Austein, Larson,
64 Massey, and Rose; March 2005. <https://www.ietf.org/rfc/rfc4035.txt>

65 *Lightweight Directory Access (LDAP) Protocol*, RFC 4511, Sermersheim, Ed., June 2006. <https://tools.ietf.org/html/rfc4511>

67 *Automated Updates of DNS Security (DNSSEC) Trust Anchors*, RFC 5011, StJohns, September
68 2007. <https://tools.ietf.org/html/rfc5011>

69 *The Transport Layer Security (TLS) Protocol Version 1.2*, RFC 5246, Dierks and Rescorla, August,
70 2008. <https://tools.ietf.org/html/rfc5246>

71 *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*;
72 Proposed Standard; IETF RFC 5280; Cooper, Santesson, Farrell, Boeyen (Entrust),
73 Housley, Polk; May 2008. <https://datatracker.ietf.org/doc/rfc5280/>

74 *Simple Mail Transfer Protocol*, IETF RFC 5321, Draft Standard, Kleinstein, October 2008. <https://tools.ietf.org/html/rfc5321>

76 *Secure/Multipurpose Internet Mail Extensions (S/MIME)*, Version 3.2, Message Specification,
77 Proposed Standard, IETF RFC 5751, ISSN: 2070-1721, Ramsdell and Turner, January
78 2010. <https://tools.ietf.org/html/rfc5751>

79 *Multicast Mobility in Mobile IP Version 6 (MIPv6): Problem Statement and Brief Survey*; RFC
80 5757; Schmidt, Waehlisch, and Fairhurst; February 2010. <https://tools.ietf.org/html/rfc5757>

82 *The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS)*
83 Application (ENUM); RFC 6116; Bradner, Conroy, and Fujiwara; March 2011. <https://tools.ietf.org/html/rfc6116>

85 *Use Cases and Requirements for DNS-Based Authentication of Named Entities (DANE)*, IETF RFC
86 6394, ISSN: 2070-1721, Barnes, October 2011. <https://tools.ietf.org/html/rfc6394>

87 *The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security Protocol:*
88 *TLSA*, Proposed Standard, IETF RFC 6698, ISSN: 2070-1721, Hoffman and Schlyter,
89 August 2012. <https://tools.ietf.org/html/rfc6698>

90 *Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation*
91 *List (CRL) Profile*, Proposed Standard, IETF RFC 6818, ISSN: 2070- 1721, Yee, January
92 2013. <https://tools.ietf.org/html/rfc6818>

93 *Adding Acronyms to Simplify Conversations about DNS-Based Authentication of Named Entities*
94 *(DANE)*, RFC 7218, Gudmundsson, April 2014. <https://tools.ietf.org/html/rfc7218>

95 *The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational*
96 *Guidance*, RFC 7671, Dukhovni and Hardaker, October 2015. <https://tools.ietf.org/html/rfc7671>

98 *SMTP security via opportunistic DANE TLS*, RFC 7672, Dukhovni and Hardaker, May 26, 2015.
99 <https://tools.ietf.org/html/rfc7672>

100 *Using Secure DNS to Associate Certificates with Domain Names For S/MIME*, IETF Internet Draft
101 Work in Progress, draft-ietf-dane-smime-12, Hoffman and Schlyter, July 31, 2016.
102 <https://datatracker.ietf.org/doc/draft-ietf-dane-smime/>

103 *Domain Name System-Based Security for Electronic Mail*, Barker, National Institute of Standards
104 and Technology's Dakota Consulting IDIQ Contract SB1341-12-CQ-0011, Task Order 15-
105 421 Task 3 Report #2, December 17, 2016. https://nccoe.nist.gov/sites/default/files/library/NCCoE_DNS-Based_Secure_E-Mail_BB.pdf

107 *How To Set Up a Postfix E-Mail Server with Dovecot*, DigitalOcean, November 14, 2013. <https://www.digitalocean.com/community/tutorials/how-to-set-up-a-postfix-e-mail-server-with-dovecot>

110 "Setting Postfix to encrypt all traffic when talking to other mailservers," *Snapdragon Tech Blog*,
111 August 9, 2013. <http://blog.snapdragon.cc/2013/07/07/setting-postfix-to-encrypt-all-traffic-when-talking-to-other-mailservers/>

1 Appendix C Platform Operation and 2 Observations

3 C.1 Operations Scenarios

4 Both server-to-server encryption (Scenario 1) and user signature (Scenario 2) of electronic mail
5 are demonstrated. Demonstrations of the security platform include attempts by fraudulent
6 actors to pose as the originator of email and man-in-the-middle attackers attempting to disrupt
7 the validation the S/MIME signature. Events are included that involve all components and
8 demonstrate that each of the MUAs can be used with both MTAs, and both MTAs can run with
9 each of the four DNS stacks. Use of self-signed certificates and of certificates from local and
10 well-known certificate authorities are included. The events do not cover all possible
11 combinations of components for both mail origination and receipt, but they do include
12 demonstration of both Exchange and Postfix as senders, all four DNS services, and both
13 Exchange and Postfix as recipients accessed by both Outlook and Thunderbird MUAs. For each
14 event identified below, we identify the components involved, operator actions required by both
15 the sender and the receiver, and observed results. For purposes of avoiding excessive repetition
16 in test events, each event includes demonstration of both scenarios.

17 C.1.1 Server-to-Server Encrypted Email in Scenario 1

18 An individual needed to enter into an email exchange with an individual in another organization
19 that required protected transfer of information. Each individual exchanged email via the
20 respective parent organizations' mail servers. Users connected to their organizations' respective
21 mail servers within a physically protected zone of control.

22 The policy of the parent organizations required encryption of the information being exchanged.
23 The security afforded by the cryptographic process was dependent on the confidentiality of
24 encryption keys from unauthorized parties. The mail servers were configured to use X.509
25 certificates to convey keying material during an encryption key establishment process.

26 DNSSEC was employed to ensure that each sending mail server connected to the legitimate and
27 authorized receiving mail server from which its X.509 certificate was obtained. DANE resource
28 records were employed to bind the cryptographic keying material to the appropriate server
29 name. STARTTLS was employed to negotiate the cryptographic algorithm to be employed with
30 TLS in the email exchange in which the message was transferred. Encryption of the email
31 message was accomplished by the originator's email server, and decryption of the email
32 message was accomplished by the recipient's email server.

33 C.1.2 Signed Email in Scenario 2

34 Scenario 2 supports the case of an individual needing to enter into an exchange of email that
35 requires integrity protection with an individual in another organization that. Each individual
36 exchanged email via the respective parent organizations' mail servers. Users connected to their
37 organizations' respective mail servers within a physically protected zone of control.

38 The policies of the parent organizations required cryptographic digital signature of the message
 39 to provide integrity protection and source authentication of the email message. S/MIME is a
 40 widely available and used protocol for digitally signing electronic mail. Each organization
 41 therefore generated X.509 certificates for their users that included the public portion of their
 42 signature keys. These certificates were then published in the DNS using the appropriate DANE
 43 DNS Resource Record (RR) type.

44 DNSSEC was used to provide assurance that the originating user's mail server connected to the
 45 intended recipient's mail server. DANE records were employed to bind the cryptographic
 46 certificates to the appropriate server (for TLS) and individual user (for S/MIME), respectively.
 47 TLS was employed to provide confidentiality. Digital signature of the email message was
 48 accomplished by the originator's email client. Validating the signature (hence the integrity of
 49 the authorization provided in the email message) was accomplished by the recipient's email
 50 client.

51 **C.1.3 Handling of Email from Fraudulent Sender**

52 Demonstrations of the security platform in both scenarios included an attempt by a fraudulent
 53 actor to pose as the originator of the email. Where it was implemented, DANE was used to
 54 expose the fraudulent originator's attempt.

55 **C.1.4 Handling of Man-in-the-Middle Attack**

56 Demonstration of the security platform in both scenarios also included a man-in-the-middle
 57 attacker attempting to disrupt the validation of the S/MIME signature. Where DANE was
 58 implemented, the attempts were shown to fail due to use of DNSSEC and DANE records.

59 **C.1.5 Effects of DNS Errors**

60 A DANE-enabled Postfix MTA sent message traffic to four Exchange MTAs with one
 61 Authoritative Server serving all four zones. An NSD4 Authoritative DNS server and Unbound
 62 recursive server was provided for the Postfix MTA, and a Secure64 DNS Authority and Signer
 63 provided the DNS services for the Exchange zones.

64 **C.2 Test Sequences**

65 The test and demonstration events selected were chosen to demonstrate the functionality
 66 available in both scenarios, the effectiveness of available DNS services, and the interoperability
 67 of components. The event selection objectives also included keeping the events to a
 68 manageable number, while capturing significant performance information. As a result, several
 69 stacks of contributed MUA, MTA, and DNS service components were demonstrated in the
 70 NCCoE laboratory environment, and representative NCCoE laboratory configurations were
 71 shown exchanging email with two different external sites using several cryptographic certificate
 72 types (certificates from Well-Known CAs (with TLSA RR cert usage (CU) of 1), Enterprise CAs
 73 (with TLSA RR cert usage of 2), and Self-Signed CAs (with TLSA RR cert usage of 3). The first
 74 external site used Secure64 DNS services, a Postfix MTA, and a Thunderbird MUA with an Apple

75 Keychain Utility. The second external site used NLnet Labs DNS services, a Postfix MTA, and a
 76 Thunderbird MUA.

77 **C.2.1 Test Sequence 1: MUA/MTA/DNS Service Combinations**
 78 **Exchanged Signed and Encrypted Email with a Secure64 Site and**
 79 **an NLnet Labs Site**

80 An Outlook MUA, interfacing with an Exchange MTA, was configured to use Active Directory
 81 and BIND DNS services in turn. Each of the six configurations exchanged email with 1) a
 82 Secure64 MUA/MTA/DNS service stack that included a Postfix MTA and a Thunderbird MUA
 83 running on a Mac OS System, and 2) an NLnet Labs MUA/MTA/ DNS service stack that included
 84 a Postfix MTA and a Thunderbird MUA running on Linux. The events include events showing use
 85 of Well-Known CAs (CU-1), Enterprise CAs (CU=2), and Self-Signed Certificates (CU=3) for TLS
 86 and S/MIME-enabled mail receivers and S/MIME. Digital signature of the messages was logged.
 87 All messages were S/MIME signed. Outlook attempted to verify received messages (Scenario 2).
 88 Signature verification results were noted. DNS name verification results were noted. [Figure 2.1](#)
 89 above depicts the set-up for laboratory support for the Secure64 destination variant of this test
 90 sequence.¹

91 **C.2.1.1 Active Directory and DNS Server in NCCoE Laboratory**

92 The Active Directory, DNS Server, an Exchange MTA, and an Outlook MUA were configured with
 93 appropriate certificates for each deployment scenario. These certificate policies include S/
 94 MIME and TLS certificates from a Well-Known CA, certificates from an Enterprise CA, and self-
 95 signed certificates (using TLSA and SMIMEA parameters CU=1, CU=2, and CU=3 respectively).
 96 Each of these three variations sent S/MIME signed and TLS encrypted email to a Secure64 site
 97 and an NLnet Labs site. The Secure64 site was using a MacBook-hosted Thunderbird MUA, a
 98 Postfix/Dovecot MTA, DNS Cache/DNS Authority services for processing received messages,
 99 and DNS Signer for outbound messages. The NLnet site was using an Intel-hosted Thunderbird
 100 MUA, a Postfix/Dovecot MTA, NSD4 and Unbound for processing received messages, and
 101 OpenDNSSEC for outbound messages. Each of the events included the NCCoE Laboratory
 102 configuration sending a signed and encrypted message to the remote sites, and a signed
 103 response being sent from each remote site to the NCCoE configuration.

104 **1. Event 1: Outlook MUA Using an Exchange MTA using Well-Known CA issued Certificates for**
 105 **TLS and S/MIME**

106 **Expected Outcome:** NCCoE Outlook MUA sent a test message in an S/MIME signed email
 107 using Active Directory DNS Services and a Well-Known CA (CU=1) to Secure 64 and NLnet
 108 Labs, and both recipients returned responses that were S/MIME signed. The signature for
 109 the received messages was verified.

110 **Observed Outcome:** As expected, the messages were authenticated and a log file was
 111 saved.

112 **2. Event 2: Outlook MUA Using an Exchange MTA using Enterprise CA issued Certificates for**
 113 **TLS and S/MIME**

1. The connections depicted in [Figure 2.1](#) are actually for the first Sequence 2 configuration. Capabilities for Sequence 1 support are shown as dotted lines.

Expected Outcome: NCCoE Outlook MUA sent a test message in an S/MIME signed email using Active Directory DNS Services and an Enterprise CA (CU=2) to Secure 64 and NLnet Labs, and both recipients returned responses that were S/MIME signed. The signature for the received messages was verified.

Observed Outcome: As expected, the messages were authenticated and a log file was saved.

3. **Event 3:** Outlook MUA Using an Exchange MTA using Self-Signed Certificate for TLS and S/MIME

Expected Outcome: NCCoE Outlook MUA sent a test message in an S/MIME signed email using Active Directory DNS Services and a self-signed TLS certificate (CU=3) to Secure 64 and NLnet Labs, and both recipients returned responses that were S/MIME signed. The signature for the received messages was verified.

Observed Outcome: As expected, the message was authenticated and a log file was saved.

¹²⁷ C.2.1.2 BIND in NCCoE Laboratory

The BIND DNS Server, an Exchange MTA, and an Outlook MUA were configured with appropriate certificates for each deployment scenario. These certificate policies include S/MIME and TLS certificates Well-Known CA, certificates from an Enterprise CA, and self-signed certificates (TLSA/SMIMEA parameters CU=1, CU=2, and CU=3 respectively). Each of these three variations sent S/MIME signed and TLS encrypted email to a Secure64 site and an NLnet Labs site. The Secure64 site was using a MacBook-hosted Thunderbird MUA, a Postfix/Dovecot MTA, DNS Cache/DNS Authority services for processing received messages, and DNS Signer for outbound messages. The NLnet site was using an Intel-hosted Thunderbird MUA, a Postfix/Dovecot MTA, NSD4 and Unbound for processing received messages, and OpenDNSSEC for outbound messages. Each of the events included the NCCoE Laboratory configuration sending a signed message to the remote sites, and a signed response being sent from each remote site to the NCCoE configuration.

1. **Event 4:** Outlook MUA Using an Exchange MTA using Well-Known CA issued Certificates for TLS and S/MIME

Expected Outcome: NCCoE Outlook MUA sent a test message in an S/MIME signed email using a BIND DNS Server and Well-Known CA (CU=1) issued certificates to Secure64 and NLnet Labs, and both Secure64 and NLnet Labs returned a response that was S/MIME signed. The signature for the received messages was verified.

Observed Outcome: As expected, the message was authenticated and a log file was saved.

2. **Event 5:** Outlook MUA Using an Exchange MTA using an Enterprise CA issued Certificates for TLS and S/MIME

Expected Outcome: NCCoE Outlook MUA sends a test message in an S/MIME signed email using a BIND DNS Server and an Enterprise CA (CU=2) issued certificates to Secure64 and NLnet Labs, and both Secure64 and NLnet Labs returned a response that was S/MIME signed. The signature for the received messages was verified.

Observed Outcome: As expected, the message was authenticated and a log file was saved.

3. **Event 6:** Outlook MUA Using an Exchange MTA using Self-Signed Certificates for TLS and S/MIME

156 **Expected Outcome:** NCCoE Outlook MUA sent a test message in an S/MIME signed email
 157 using a BIND DNS Server and self-signed certificates (CU=3) to Secure64 and NLnet, and
 158 both Secure64 and NLnet returned a response that was S/MIME signed. The signature for
 159 the received messages was verified.

160 **Observed Outcome:** As expected, the message was authenticated and a log file was saved.

161 C.2.2 Test Sequence 2: MUA/MTA/DNS Service Combinations

162 Exchanged Signed and Encrypted Email with an NLnet Labs Site 163 and a Secure64 Site

164 Outlook and Thunderbird MUAs, configured to use a Postfix MTA with Dovecot IMAP support,
 165 were configured in turn to use BIND and Secure64's DNS Authority, DNS Cache, and DNS Signer
 166 implementations. Each of the six configurations exchanged email with a Secure64 site that
 167 included a Thunderbird MUA, DNS Cache/DNS Signer/DNS Authority DNS services, and Postfix/
 168 Dovecot MTA and an NLnet Labs MUA/MTA/ DNS service stack that included a Thunderbird
 169 MUA, NSD4, Unbound, and OpenDNSSEC DNS services and a Postfix/Dovecot MTA. The test
 170 events include using Well-Known CA issued (TLSA/SMIMEA CU=1), Enterprise CA issued (CU=2),
 171 and Self-Signed Certificates (CU=3). Email messages between MTAs were encrypted and
 172 successfully decrypted. (Scenario 1). Signature and encryption were logged. All messages were
 173 S/MIME signed. Outlook attempted to verify received messages (Scenario 2). Signature
 174 verification results were noted. DNS name verification results were noted. Figure 2 above
 175 depicts the set-up for laboratory support for this test sequence, with connections selected for
 176 Event 7 below.

177 C.2.2.1 BIND and Postfix/Dovecot in NCCoE Laboratory

178 Outlook, then Thunderbird mail clients were configured to use Postfix/Dovecot MTAs and BIND
 179 DNS servers. Each of these three configurations sent S/MIME signed and TLS encrypted email to
 180 a Secure64 site and an NLnet Labs site. The Secure64 site was using a Thunderbird MUA using
 181 Secure64's Apple Key Chain Utility tool that allows a host to obtain X.509 certificates via of
 182 DANE RRs, DNS Cache/DNS Signer/DNS Authority DNS services, and a Postfix/Dovecot MTA for
 183 mail. The NLnet Labs site was using a Thunderbird MUA, a Postfix/Dovecot MTA, and NSD4,
 184 Unbound, and OpenDNSSEC DNS Services. Each of the three events included the NCCoE
 185 Laboratory configuration sending a S/MIME signed and TLS encrypted message to the Secure64
 186 and NLnet Labs sites, and signed and encrypted responses being sent from the Secure64 and
 187 NLnet Labs site to the NCCoE.

188 1. **Event 7:** Outlook MUA Using a Postfix/Dovecot MTA and Well-Known CA Issued Certificates
 189 for TLS and S/MIME

190 **Expected Outcome:** NCCoE Outlook MUA using BIND for DNS sent a test message in an S/
 191 MIME signed email to Secure64 and NLnet Labs. Secure64 and NLnet Labs returned
 192 responses that were S/MIME signed and TLS encrypted. The received messages were
 193 successfully decrypted, and the signatures were verified. All S/MIME and MTA TLS
 194 certificates in this test were issued from a well-known CA and TLSA/SMIMEA RR Certificate
 195 Usage parameter set to 1.

196 **Observed Outcome:** As expected, the message was authenticated and decrypted, and a log
 197 file was saved.

198 2. **Event 8:** Thunderbird MUA Using a Postfix/Dovecot MTA and Enterprise CA Issued
 199 Certificates for TLS and S/MIME

200 **Expected Outcome:** NCCoE Thunderbird MUA using BIND for DNS sent a test message in an
 201 S/MIME signed email to Secure64 and NLnet Labs. Secure64 and NLnet Labs returned
 202 responses that were S/MIME signed and TLS encrypted. The received messages were
 203 successfully decrypted, and the signatures were verified. All S/MIME and MTA TLS
 204 certificates in this test were issued from an enterprise local CA and TLSA/SMIMEA RR
 205 Certificate Usage parameter set to 2.

206 **Observed Outcome:** As expected, the message was authenticated and decrypted. and a log
 207 file was saved.

208 3. **Event 9:** Thunderbird MUA Using a Postfix/Dovecot MTA and Self-Signed Certificates

209 **Expected Outcome:** NCCoE Thunderbird MUA using BIND for DNS sent a test message in an
 210 S/MIME signed email to Secure64 and NLnet Labs. Secure64 and NLnet Labs returned
 211 responses that were S/MIME signed and TLS encrypted. The received messages were
 212 successfully decrypted, and the signatures were verified. All S/MIME and MTA TLS
 213 certificates in this test were self-signed and TLSA/SMIMEA RR Certificate Usage parameter
 214 set to 3.

215 **Observed Outcome:** As expected, the message was authenticated and decrypted, and a log
 216 file was saved.

217 **C.2.2.2 Postfix/Dovecot with DNS Authority, DNS Cache, and DNS Signer in NCCoE
 218 Laboratory**

219 A Thunderbird client was configured to use DNS Authority, DNS Cache, and DNS Signer Servers
 220 and use a Postfix/Dovecot MTA. Each of these three configurations sent S/MIME signed and TLS
 221 encrypted email to a Secure64 site and an NLnet Labs site. The Secure64 site was using a
 222 Thunderbird MUA that employed Secure64's Apple Key Chain Utility tool that allows a host to
 223 obtain X.509 certificates via of DANE RRs, DNS Cache/DNS Signer/DNS Authority DNS services,
 224 and a Postfix/ Dovecot MTA for mail. The NLnet Labs site was using a Thunderbird MUA, a
 225 Postfix/Dovecot MTA, and NSD4, Unbound, and OpenDNSSEC DNS Services. Each of the three
 226 events included the NCCoE Laboratory configuration sending an S/MIME signed and TLS
 227 encrypted message to the Secure64 and NLnet Labs sites, and signed and encrypted responses
 228 being sent from the Secure64 and NLnet Labs site to the NCCoE.

229 1. **Event 10:** Thunderbird MUA Using a Postfix/Dovecot MTA and Well-Known CA Issued
 230 Certificates for TLS and S/MIME

231 **Expected Outcome:** NCCoE Thunderbird MUA using DNS Authority/Cache/Signer DNS
 232 Services and a Postfix MTA sent a test message in an S/MIME signed email to Secure64 and
 233 NLnet Labs. Secure64 and NLnet Labs returned that a message that we had S/MIME signed
 234 and TLS encrypted. The received messages were successfully decrypted, and the signatures
 235 were verified. All certificates in this test were issued from a well-known CA and TLSA/
 236 SMIMEA RR Certificate Usage parameter set to 1.

237 **Observed Outcome:** As expected, the message was authenticated and decrypted, and a log
 238 file was saved.

239 2. **Event 11:** Thunderbird MUA Using a Postfix/Dovecot MTA and Enterprise CA Issued
 240 Certificates for TLS and S/MIME

241 **Expected Outcome:** NCCoE Thunderbird MUA using DNS Authority/Cache/Signer DNS
 242 Services and a Postfix MTA sent a test message in an S/MIME signed email to Secure64 and
 243 NLnet Labs. Secure64 and NLnet Labs returned a message that we had S/MIME signed and
 244 TLS encrypted. The received messages were successfully decrypted, and the signatures
 245 were verified. All certificates in this test were issued from an enterprise CA and TLSA/
 246 SMIMEA RR Certificate Usage parameter set to 2.

247 **Observed Outcome:** As expected, the message was authenticated and decrypted, and a log
 248 file was saved.

249 3. **Event 12:** Thunderbird MUA Using a Postfix/Dovecot MTA and Self-Signed Certificates for
 250 TLS and S/MIME

251 **Expected Outcome:** NCCoE Thunderbird MUA using DNS Authority/Cache/Signer DNS
 252 Services and a Postfix MTA sent a test message in an S/MIME signed email to Secure64 and
 253 NLnet Labs. Secure64 and NLnet Labs returned a message that we had S/MIME signed and
 254 TLS encrypted. The received messages were successfully decrypted, and the signatures
 255 were verified. All certificates in this test were self-signed and TLSA/SMIMEA RR Certificate
 256 Usage parameter set to 3.

257 **Observed Outcome:** As expected, the message was authenticated and decrypted, and a log
 258 file was saved.

259 C.2.3 Sequence 3: Fraudulent DNS Address Posing as Valid DNS 260 Address Contacting Recipient MTAs

261 Fraudulently S/MIME signed email was sent from a malicious sender to recipients using
 262 Outlook and Thunderbird MUAs configured to use Exchange and Postfix as MTAs. The Outlook/
 263 Exchange configuration used Active Directory as its DNS server. The configurations employing
 264 Postfix/Dovecot MTAs were demonstrated with each of the other three contributed DNS
 265 Services. In one event, the Thunderbird MUA employed an Apple Key Chain Utility tool that
 266 allows a host to obtain X.509 certificates via of DANE RRs. All events were conducted using well-
 267 known CA and Enterprise CA-issued certificates for the impersonated sender. The fraudulent
 268 site attempted to spoof a valid sending domain belonging to a Secure64 site that was
 269 configured with DNS Authority/Cache/Signer DNS services, a Postfix/Dovecot MTA, and
 270 Thunderbird¹ equipped with the Apple Key Chain utility. An Outlook/Exchange/Active Directory
 271 set-up acted as the fraudulent site. The email exchange between organizations was carried over
 272 TLS, and the email message was S/MIME signed on the fraudulent users' client device. The set-
 273 up for this sequence is depicted in [figure C.1](#) below.

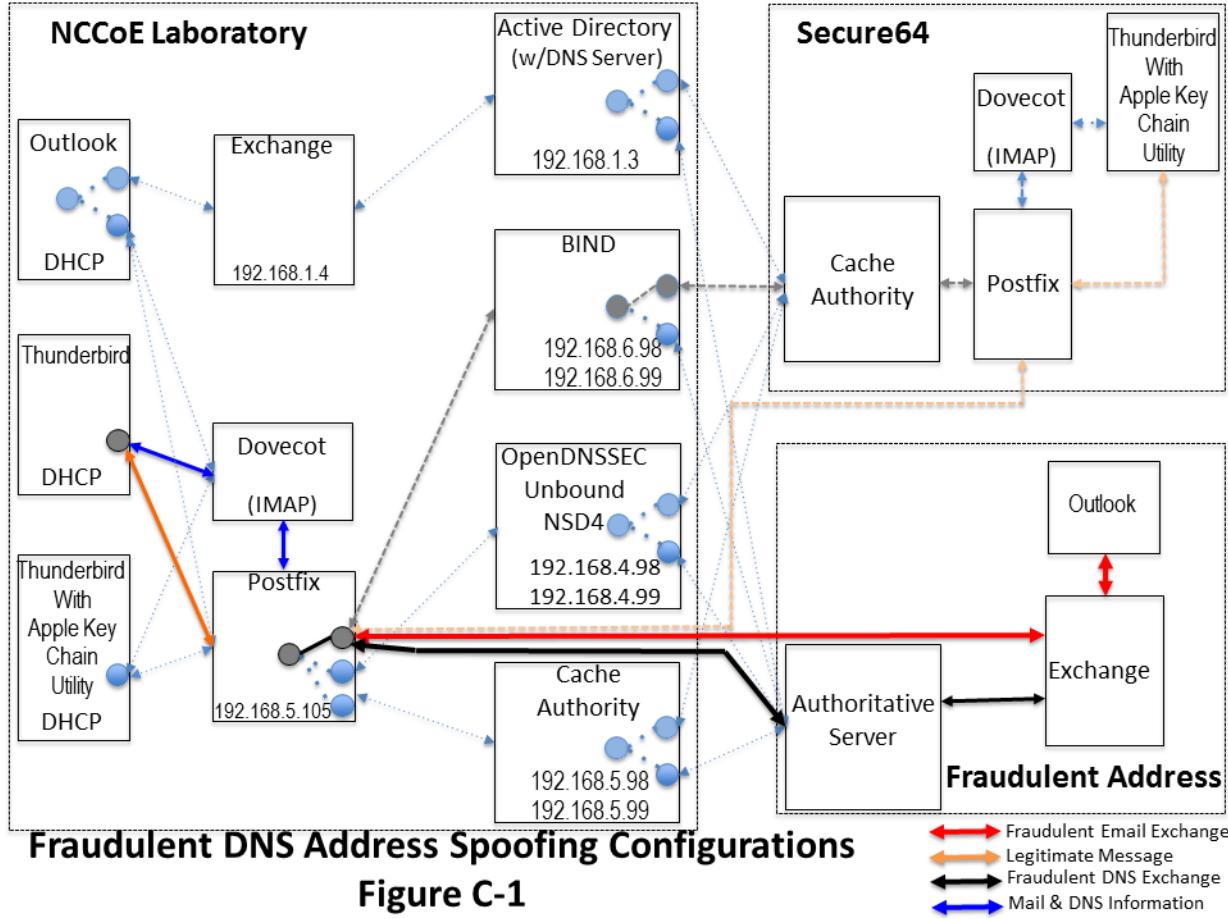
274 C.2.3.1 Spoofing Attempts Against Exchange and Postfix/Dovecot Configurations Using 275 Enterprise CA Issued Certificates (CU=1)

276 The target set-up is comprised of (alternatively): Active Directory and DNS Server, BIND DNS
 277 Server, NLnet Labs DNS Services, and Secure64 DNS services with Microsoft Outlook/Exchange,
 278 Outlook/Postfix/Dovecot, and Thunderbird/Postfix/ Dovecot mail configurations. For purposes
 279 of this demonstration, two certificates were issued for each domain. One of these was valid and
 280 published as a DNSSEC signed SMIMEA RR in the target's zone. The second (spoofed) certificate

1. Technically, this shouldn't matter. Secure64 isn't sending the mail, so the MUA isn't involved.

281 is not in the DNS. The fraudulent site possessed the spoofed certificates and, posing as a valid Secure64 site, attempted to send emails to
 282 the NCCoE Laboratory target configurations. The email and DNS transactions were logged in each case, and the results are provided
 283 below.

284 **Figure C.1 Fraudulent DNS Address Spoofing Configurations**



285 **1. Event 13: Outlook MUA, Exchange MTA, and Active Directory DNS Services**

287 **Expected Outcome:** Using S/MIME, Outlook validated the message from the attacker (as
288 DANE is not enabled in Outlook at this time).

289 **Observed Outcome:** As expected and a log file was saved.

290 2. **Event 14:** Thunderbird MUA, Postfix/Dovecot MTA and NLnet Labs DNS Services

291 **Expected Outcome:** Using S/MIME and DANE, Thunderbird recognizes that the certificate
292 has not been validated and does not deliver the message to the user. Thunderbird will flag
293 the signature as invalid.

294 **Observed Outcome:** As expected and a log file was saved.

295 3. **Event 15:** Thunderbird MUA, Postfix/Dovecot MTA and Secure64 DNS Services

296 **Expected Outcome:** Using S/MIME and DANE, Thunderbird with the Apple Key Chain Utility
297 recognizes that the certificate has not been validated and does not deliver the message to
298 the user.

299 **Observed Outcome:** As expected and a log file was saved.

300 C.2.3.2 **Spoofing Attempts Against Exchange and Postfix/Dovecot Configurations Using Self-
301 Signed Certificates (CU=3)**

302 The target set-up is configured to use Active Directory with Outlook and Exchange; and in a
303 separate set of tests: BIND and NLnet Labs DNS Services (alternatively) were configured with a
304 Thunderbird MUA and a Postfix/Dovecot MTA. The fraudulent site, posing as a valid Secure64
305 site, attempted to send an email to the NCCoE Laboratory target. The email and DNS
306 transactions were logged in each case, and the results are provided below.

307 1. **Event 16:** Postfix MTA Using an Active Directory DNS Service

308 **Expected Outcome:** Using only S/MIME, Outlook will fail to validate the message from the
309 attacker as it was signed by an untrusted root, but not marked as a possible attack.

310 **Observed Outcome:** As expected and a log file was saved.

311 2. **Event 17:** Postfix MTA Using a BIND DNS Service

312 **Expected Outcome:** Using S/MIME and DANE, Thunderbird with the Apple Key Chain Utility
313 recognizes that the certificate has not been validated and does not deliver the message to
314 the user.

315 **Observed Outcome:** As expected and a log file was saved.

316 3. **Event 18:** Postfix MTA Using an NLnet DNS Service

317 **Expected Outcome:** Using S/MIME and DANE, Thunderbird with the Apple Key Chain Utility
318 recognizes that the certificate has not been validated and does not deliver the message to
319 the user.

320 **Observed Outcome:** As expected and a log file was saved.

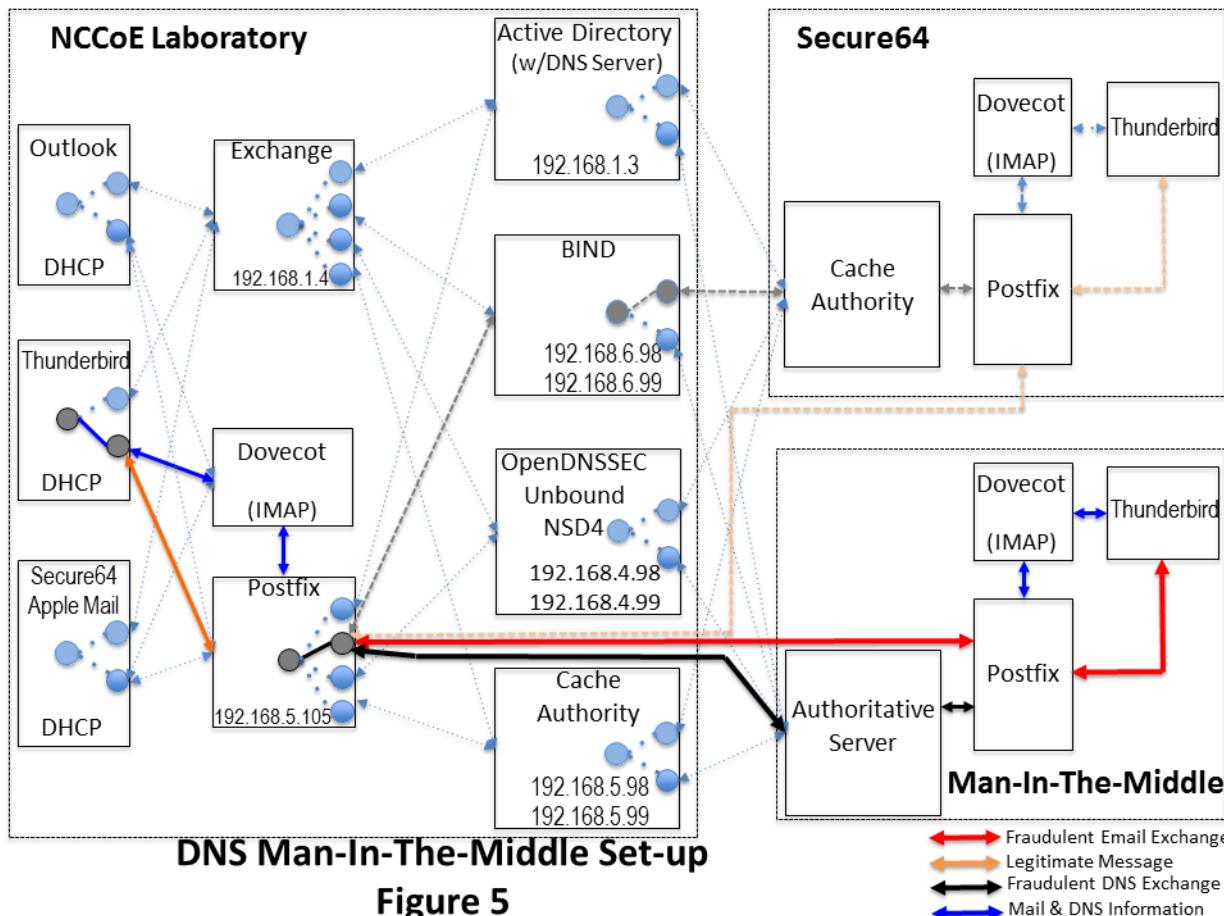
321 C.2.4 Sequence 4: Man-in-the-Middle Attack on Postfix-to-Postfix 322 Connection

323 An NCCoE system attempted to send a TLS protected email from Exchange and Postfix MTAs (in
324 turn) to an external Postfix MTA using DNS Authority/Cache/Signer for DNS services. The NCCoE
325 Exchange MTA used Active Directory DNS Services, and the Postfix/Dovecot MTA used BIND and
326 NSD4/Unbound/OpenDNSSEC DNS services. A S/MIME signed email was sent to an external
327 Postfix MTA. Four events were conducted using Well-Known CA issued certificates, four events
328 were conducted using Enterprise CA issued certificates (TLSA/SMIMEA RR parameter of CU=2)
329 for TLS and S/MIME on the receiver side, and three events were conducted using self-signed
330 certificates (TLSA/SMIMEA RR parameter of CU=3) for TLS and S/MIME on the receiver side. An
331 Outlook/Exchange/Active Directory stack acted as a man-in-the-middle and attempted to
332 intercept the message. [Figure C.2](#) depicts the configuration for a man-in-the-middle
333 demonstration. Note that the sender is being misdirected to a malicious email server only. This
334 is to simulate a lower level attack where email is sent (via route hijacking or similar low level
335 attack) to a Man-in-the-Middle. [Figure C.2](#) depicts the configurations used with the
336 Thunderbird/Postfix/Dovecot/Bind option selected.

337 C.2.4.1 Man-in-the-Middle Attack when Senders and Receivers use Well-Known CA Issued 338 Certificates (CU=1)

339 The sender set-up was comprised of Active Directory and DNS Server, BIND DNS Service, or
340 NLnet Labs DNS Services with Outlook and Thunderbird MUAs using an Exchange MTA. In the
341 fourth event, the sender is a Thunderbird MUA with a Secure64 Apple Key Chain utility utilizing
342 NSD4/Unbound/OpenDNSSEC DNS services and a Postfix/Dovecot MTA. Enterprise CA issued
343 certificates are used on the receiver side for TLS. Each of the four configurations attempts to
344 initiate an email exchange with an external Secure64 site. The man-in-the-middle, an Outlook/
345 Exchange/Active Directory stack, attempts to spoof the intended receiver and accept the email.
346 The email and DNS transactions were logged in each case, and the results are provided below.

Figure C.2 Man-in-the-Middle Event Configurations



1. Event 19: Outlook MUA, Exchange MTA, and Active Directory DNS Service as Sender

Expected Outcome: The sending MTA fails to detect the spoofing. The mail connection to the MTA is established and mail is transferred.

Observed Outcome: As expected and a log file was saved.

353 2. **Event 20:** Thunderbird MUA, Exchange MTA, and BIND DNS Service as Sender

354 **Expected Outcome:** The sending MTA fails to detect the spoofing. The mail connection to

355 the MTA is established and mail is transferred.

356 **Observed Outcome:** As expected and a log file was saved.

357 3. **Event 21:** Thunderbird MUA, Postfix MTA and NSD4/Outbound/ OpenDNSSEC DNS Services

358 as Sender

359 **Expected Outcome:** The MUA using a SMIMEA utility was able to detect the fraudulent

360 email and mark the email as not validated.

361 **Observed Outcome:** As expected and a log file was saved.

362 4. **Event 22:** Thunderbird MUA with Secure64 Apple Key Chain Utility, Postfix/Dovecot MTA

363 and DNS Authority/Cache/Signer DNS Services

364 **Expected Outcome:** The MUA using a SMIMEA utility was able to detect the fraudulent

365 email and mark the email as not validated.

366 **Observed Outcome:** As expected and a log file was saved.

367 **C.2.4.2 Man-in-the-Middle Attack when Senders and Receivers use Enterprise CA Issued**

368 **Certificates (CU=2)**

369 The sender set-up was composed of Active Directory and DNS Server, BIND DNS Service, or

370 NLnet Labs DNS Services with Outlook and Thunderbird MUAs using an Exchange MTA. In the

371 fourth event, the sender is a Thunderbird MUA with a Secure64 Apple Key Chain utility utilizing

372 NSD4/Unbound/OpenDNSSEC DNS services and a Postfix/Dovecot MTA. Enterprise CA issued

373 certificates are used on the receiver side for TLS. Each of the four configurations attempts to

374 initiate an email exchange with an external Secure64 site. The man-in-the-middle, an Outlook/

375 Exchange/Active Directory stack, attempts to spoof the intended receiver and accept the email.

376 The email and DNS transactions were logged in each case, and the results are provided below.

377 1. **Event 23:** Outlook MUA, Exchange MTA, and Active Directory DNS Service as Sender.

378 **Expected Outcome:** The sending MTA fails to detect the spoofing. The mail connection to

379 the MTA is established and mail transferred.

380 **Observed Outcome:** As expected and a log file was saved.

381 2. **Event 24:** Thunderbird MUA, Exchange MTA, and BIND DNS Service as Sender.

382 **Expected Outcome:** The sending MTA fails to detect the spoofing. The mail connection to

383 the MTA is established and mail transferred.

384 **Observed Outcome:** As expected and a log file was saved.

385 3. **Event 25:** Thunderbird MUA, Postfix MTA and NSD4/Outbound/ OpenDNSSEC DNS Services

386 as Sender

387 **Expected Outcome:** The Postfix MTA detects the spoofing and closes the SMTP connection

388 before the email is sent.

389 **Observed Outcome:** As Expected.

390 4. **Event 26:** Thunderbird MUA with Secure64 Apple Key Chain Utility, Postfix/Dovecot MTA

391 and DNS Authority/Cache/Signer DNS Services

392 **Expected Outcome:** The postfix MTA detects the spoofing and closes the SMTP connection
 393 before the email is sent.

394 **Observed Outcome:** As Expected.

395 C.2.4.3 Man-in-the-Middle With Self-Signed Certificates (CU=3)

396 The sender uses an Outlook and Thunderbird MUAs sending mail through a Postfix/Dovecot
 397 MTA and using (in turn): Active Directory and DNS Server, BIND DNS Server, and NLnet Labs DNS
 398 Services. Self-signed certificates are used on the legitimate receiver side (TLSA RR parameter
 399 CU=3) for TLS. Each of the three configurations attempts to initiate an email exchange with an
 400 external Secure64 site. The man-in-the-middle, an Outlook/Exchange/ Active Directory stack,
 401 attempts to intercept the email from the NCCoE Laboratory Configuration by acting as a Man-
 402 in-the-Middle. The email and DNS transactions were logged in each case, and the results are
 403 provided below.

404 1. **Event 27:** Postfix MTA Using an Active Directory DNS Service

405 **Expected Outcome:** TLSA detects spoofing. The mail connection to the MTA is established
 406 but breaks before the mail is transferred.

407 **Observed Outcome:** As expected and a log file was saved.

408 2. **Event 28:** Thunderbird MUA, Exchange MTA, and BIND DNS Service

409 **Expected Outcome:** Exchange fails to detect the man-in-the-middle and sends the email.

410 **Observed Outcome:** As expected and a log file was saved.

411 3. **Event 29:** Thunderbird MUA with Secure64 Apple Key Chain Utility, Exchange MTA and
 412 NSD4/Outbound/OpenDNSSEC DNS Services

413 **Expected Outcome:** Exchange fails to detect the man-in-the-middle and sends the email.

414 **Observed Outcome:** As expected and a log file was saved.

415 C.2.5 Sequence 5: Effects of DANE Errors

416 In Sequence 5, A DANE-enabled Postfix MTA sent message traffic to four other postfix MTAs.
 417 See [figure C.3](#). A single BIND instance was set up to serve the TLSA and A RRs for the four
 418 receivers. One of the receiving MTAs did not employ DANE. The second employed DANE with a
 419 valid TLSA with the certificate usage field¹ set to 3. The third employed a TLSA with a certificate
 420 usage field of 2, but with an incomplete (i.e. bad) PKI certification path (generating a PKIX
 421 validation failure). The TLSA contained a local enterprise trust anchor, but the server did not
 422 have the full certificate chain (missing intermediate certificate). The final one employed DANE
 423 with a TLSA RR using Certificate Usage of 3, but there was a mismatch between the server cert
 424 and TLSA RR (generating a DANE validation failure).

1. RFC 6698, The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA, Section 2.1.1. <https://tools.ietf.org/html/rfc6698#section-2.1.1>

425 C.2.5.1 Event 30: DNS/DANE Error Results

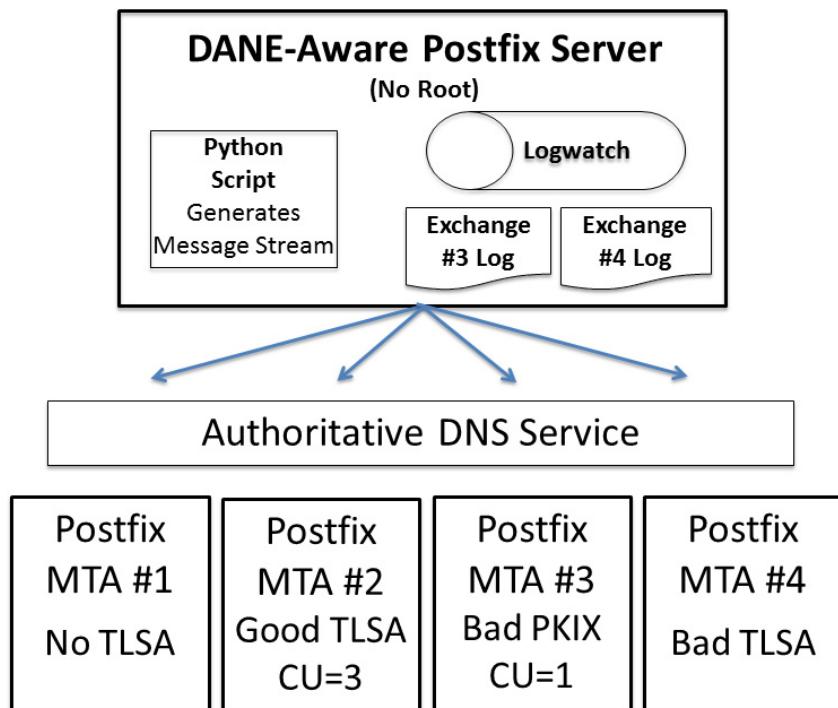
426 The test sequence was set up as described above. The sending MTA was set with different TLS
 427 and DANE requirements configuration. Postfix can be configured for different “levels” of TLS
 428 and DANE processing and reliance. In the Postfix configuration file (**main.cf**) the option to turn
 429 on DANE processing is:

430 `smtp_tls_security_level = none | may | encrypt | dane | dane-only | fingerprint`
 431 `| verify | secure`

432 For this test, only **none**, **may**, **dane** and **dane-only** are relevant. These values affect how postfix
 433 establish and use TLS when sending email:

- 434 ■ **none**: The sender does not use TLS even when offered or available. Email is always sent in
 435 plaintext.
- 436 ■ **may**: The sender uses TLS opportunistically when available. No effort will be made to
 437 validate the server peer certificate, but will be used regardless.
- 438 ■ **encrypt**: The sender will only send mail when TLS is available, even if the server peer
 439 certificate is on validated. If STARTTLS is not offered, mail is deferred.
- 440 ■ **dane**: The sender attempts to use TLS when offered, and queries for TLSA RRs to help
 441 validate the server peer certificate. Mail is still sent if the validation fails, so this is
 442 sometimes referred to as “opportunistic DANE”.
- 443 ■ **dane-only**: The sender only sends mail when TLS is offered, and there is a valid TLSA RR
 444 found. Otherwise, mail is deferred.

445 **Figure C.3 Failed Delivery Logs**



447 **Expected Outcome:** Little or nothing appears in the sender's logs for messages sent to either
 448 the MTA not employing TLS or the employing a valid TSLA. The growth rates for logs for the MTA
 449 that employs a TSLA with a certificate usage field of 1, but with a PKIX failure and the one that
 450 employs mismatched server cert/TSLA (i.e. DANE validation failure) are measured.

451 **Observed Outcome:** The delivery of the email depended on the TLS/DANE status of the
 452 receiver and the TLS/DANE configuration on the sender. The results were:

453 **Table C.1 Transaction Results Based on Sender TLS/DANE Connection**

TLS/DANE Option	Receiver TLS/DANE deployment			
	No TLS	TLS with valid DANE RR	TLS with DANE PKIX failure	TLS with DANE TSLA RR Error
none	Mail sent in plaintext	Mail sent in plaintext	Mail sent in plaintext	Mail sent in plaintext
may	Mail sent in plaintext	Mail sent over anonymous TLS (i.e., no validation of certificate)	Mail sent over anonymous TLS (i.e., no validation of certificate)	Mail sent over anonymous TLS (i.e., no validation of certificate)
dane	Mail sent in plaintext	Mail sent over TLS (with DANE validation logged)	Mail sent over anonymous TLS (i.e., no validation of certificate)	Mail sent over anonymous TLS (i.e., no validation of certificate)
dane-only	Mail not sent	Mail sent over TLS (with DANE validation logged)	Mail not sent	Mail not sent

454 From the above table, when the sender was configured to never use TLS, the mail was sent in
 455 plaintext regardless of the TLS/DANE configuration of the receiver. When the sender was
 456 configured to use TLS opportunistically, it used TLS regardless of the status of the certificate, or
 457 TSLA. In fact, the sender did not issue a query to find TSLA RRs even if published. When the
 458 sender used opportunistic DANE, it used TLS when available regardless of the DANE validations
 459 results. If validation failed, the mail was still sent and the result was logged as an **Untrusted** or
 460 **Anonymous** TLS connection, depending on the presence of a TSLA RR.

461 Of the four options used in the lab, **dane-only** was the most rigorous in what a sender will
 462 accept before sending mail. When the receiver did not offer the STARTTLS option, or lacked a
 463 TSLA RR, mail was not sent. Likewise, if a TSLA RR was present, but there was an error in
 464 validation (either the TSLA RR itself had an error, or PKIX failed), the mail was not sent.
 465 Therefore, use of this option was not recommended for general use as this resulted in the
 466 majority of email being deferred. It should only be used in scenarios where senders and
 467 receivers are coordinated and maintain a stable DANE deployment.

Appendix D Secure Name System (DNS) Deployment Checklist

The following checklist includes actions recommended by NIST SP 800-81-2, *Secure Domain Name System (DNS) Deployment Guide*. The checklist provides secure deployment guidelines for each DNS component based on policies and best practices. The primary security specifications (with associated mechanisms) on which the checklist is based are as follows:

- Internet Engineering Task Force (IETF) Domain Name System Security Extensions (DNSSEC) specifications, covered by Request for Comments (RFC) 3833, 4033, 4034, and 4035
- IETF Transaction Signature (TSIG) specifications, covered by RFCs 2845 and 3007

While not all of the checklist recommendations apply to all cases of DNS-protected email security, the checklist is a reliable guide for secure deployment of DNS components.

1. **Checklist item 1:** When installing the upgraded version of name server software, the administrator should make necessary changes to configuration parameters to take advantage of new security features.
2. **Checklist item 2:** Whether running the latest version or an earlier version, the administrator should be aware of the vulnerabilities, exploits, security fixes, and patches for the version that is in operation in the enterprise. The following actions are recommended (for BIND deployments):
 - Subscribe to ISC's mailing list called **bind-announce** or **nsd-users** for NSD
 - Periodically refer to the BIND vulnerabilities page at <http://www.isc.org/products/BIND/bind-security.html>
 - Refer to CERT/CC's Vulnerability Notes Database at <http://www.kb.cert.org/vuls/> and the NIST NVD metabase at <http://nvd.nist.gov>.

For other implementations (e.g., MS Windows Server), other announcement lists may exist.

3. **Checklist item 3:** To prevent unauthorized disclosure of information about which version of name server software is running on a system, name servers should be configured to refuse queries for its version information.
4. **Checklist item 4:** The authoritative name servers for an enterprise should be both network and geographically dispersed. Network-based dispersion consists of ensuring that all name servers are not behind a single router or switch, in a single subnet, or using a single leased line. Geographic dispersion consists of ensuring that not all name servers are in the same physical location, and hosting at least a single secondary server off-site.
5. **Checklist item 5:** If a hidden master is used, the hidden authoritative master server should only accept zone transfer requests from the set of secondary zone name servers and refuse all other DNS queries. The IP address of the hidden master should not appear in the name server set in the zone database.
6. **Checklist item 6:** For split DNS implementation, there should be a minimum of two physical files or views. One should exclusively provide name resolution for hosts located inside the firewall. It also can contain RRsets for hosts outside the firewall. The other file or view

40 should provide name resolution only for hosts located outside the firewall or in the DMZ,
41 and not for any hosts inside the firewall.

42 7. **Checklist item 7:** It is recommended that the administrator create a named list of trusted
43 hosts (or blacklisted hosts) for each of the different types of DNS transactions. In general,
44 the role of the following categories of hosts should be considered for inclusion in the
45 appropriate ACL:

46

- 47 • DMZ hosts defined in any of the zones in the enterprise
- 48 • all secondary name servers allowed to initiate zone transfers
- 49 • internal hosts allowed to perform recursive queries

50 8. **Checklist item 8:** The TSIG key (secret string) should be a minimum of 112 bits in length if
51 the generator utility has been proven to generate sufficiently random strings [800-57P1].
52 128 bits recommended.

53 9. **Checklist item 9:** A unique TSIG key should be generated for each set of hosts (i.e. a unique
54 key between a primary name server and every secondary server for authenticating zone
55 transfers).

56 10. **Checklist item 10:** After the key string is copied to the key file in the name server, the two
57 files generated by the dnssec-keygen program should either be made accessible only to the
58 server administrator account (e.g., root in Unix) or, better still, deleted. The paper copy of
59 these files also should be destroyed.

60 11. **Checklist item 11:** The key file should be securely transmitted across the network to name
61 servers that will be communicating with the name server that generated the key.

62 12. **Checklist item 12:** The statement in the configuration file (usually found at /etc/named.conf
63 for BIND running on Unix) that describes a TSIG key (key name (ID), signing algorithm, and
64 key string) should not directly contain the key string. When the key string is found in the
65 configuration file, the risk of key compromise is increased in some environments where
66 there is a need to make the configuration file readable by people other than the zone
67 administrator. Instead, the key string should be defined in a separate key file and referenced
68 through an include directive in the key statement of the configuration file. Every TSIG key
69 should have a separate key file.

70 13. **Checklist item 13:** The key file should be owned by the account under which the name
71 server software is run. The permission bits should be set so that the key file can be read or
72 modified only by the account that runs the name server software.

73 14. **Checklist item 14:** The TSIG key used to sign messages between a pair of servers should be
74 specified in the server statement of both transacting servers to point to each other. This is
75 necessary to ensure that both the request message and the transaction message of a
76 particular transaction are signed and hence secured.

77 15. **Checklist item 15:** Name servers that deploy DNSSEC signed zones or query signed zones
78 should be configured to perform DNSSEC processing.

79 16. **Checklist item 16:** The private keys corresponding to both the ZSK and the KSK should not
80 be kept on the DNSSEC-aware primary authoritative name server when the name server
81 does not support dynamic updates. If dynamic update is supported, the private key
82 corresponding to the ZSK alone should be kept on the name server, with appropriate
directory/file-level access control list-based or cryptography-based protections.

83 17. **Checklist item 17:** Signature generation using the KSK should be done offline, using the KSK-
 84 private stored offline; then the DNSKEY RRSet, along with its RRSIG RR, can be loaded into
 85 the primary authoritative name server.

86 18. **Checklist item 18:** The refresh value in the zone SOA RR should be chosen with the
 87 frequency of updates in mind. If the zone is signed, the refresh value should be less than the
 88 RRSIG validity period.

89 19. **Checklist item 19:** The retry value in a zone SOA RR should be 1/10th of the refresh value.

90 20. **Checklist item 20:** The expire value in the zone SOA RR should be 2 to 4 weeks.

91 21. **Checklist item 21:** The minimum TTL value should be between 30 minutes and 5 days.

92 22. **Checklist item 22:** A DNS administrator should take care when including HINFO, RP, LOC, or
 93 other RR types that could divulge information that would be useful to an attacker, or the
 94 external view of a zone if using split DNS. These RR types should be avoided if possible and
 95 only used if necessary to support operational policy.

96 23. **Checklist item 23:** A DNS administrator should review the data contained in any TXT RR for
 97 possible information leakage before adding it to the zone file.

98 24. **Checklist item 24:** The validity period for the RRSIGs covering a zone's DNSKEY RRSet should
 99 be in the range of 2 days to 1 week. This value helps reduce the vulnerability period
 100 resulting from a key compromise.

101 25. **Checklist item 25:** A zone with delegated children should have a validity period of a few
 102 days to 1 week for RRSIGs covering the DS RR for a delegated child. This value helps reduce
 103 the child zone's vulnerability period resulting from a KSK compromise and scheduled key
 104 rollovers.

105 26. **Checklist item 26:** If the zone is signed using NSEC3 RRs, the salt value should be changed
 106 every time the zone is completely resigned. The value of the salt should be random, and the
 107 length should be short enough to prevent a FQDN to be too long for the DNS protocol (i.e.
 108 under 256 octets).

109 27. **Checklist item 27:** If the zone is signed using NSEC3 RRs, the iterations value should be
 110 based on available computing power available to clients and attackers. The value should be
 111 reviewed annually and increased if the evaluation conditions change.

112 28. **Checklist item 28:** TTL values for DNS data should be set between 30 minutes (1800
 113 seconds) and 24 hours (86400 seconds).

114 29. **Checklist item 29:** TTL values for RRsets should be set to be a fraction of the DNSSEC
 115 signature validity period of the RRSIG that covers the RRset.

116 30. **Checklist item 30:** The (often longer) KSK needs to be rolled over less frequently than the
 117 ZSK. The recommended rollover frequency for the KSK is once every 1 to 2 years, whereas
 118 the ZSK should be rolled over every 1 to 3 months for operational consistency but may be
 119 used longer if necessary for stability or if the key is of the appropriate length. Both keys
 120 should have an Approved length according to NIST SP 800-57 Part 1 [800-57P1], [800-57P3].

121 **Zones that pre-publish the new public key should observe the following:**

122 31. **Checklist item 31:** The secure zone that pre-publishes its public key should do so at least
 123 one TTL period before the time of the key rollover.

124 32. **Checklist item 32:** After removing the old public key, the zone should generate a new
125 signature (RRSIG RR), based on the remaining keys (DNSKEY RRs) in the zone file.

126 33. **Checklist item 33:** A DNS administrator should have the emergency contact information for
127 the immediate parent zone to use when an emergency KSK rollover must be performed.

128 34. **Checklist item 34:** A parent zone must have an emergency contact method made available
129 to its delegated child subzones in case of emergency KSK rollover. There also should be a
130 secure means of obtaining the new KSK.

131 35. **Checklist item 35:** Periodic re-signing should be scheduled before the expiration field of the
132 RRSIG RRs found in the zone. This is to reduce the risk of a signed zone being rendered
133 bogus because of expired signatures.

134 36. **Checklist item 36:** The serial number in the SOA RR must be incremented before re-signing
135 the zone file. If this operation is not done, secondary name servers may not pick up the new
136 signatures because they are refreshed purely on the basis of the SOA serial number
137 mismatch. The consequence is that some security-aware resolvers will be able to verify the
138 signatures (and thus have a secure response) but others cannot.

139 37. **Checklist item 37:** Recursive servers/resolvers should be placed behind an organization's
140 firewall and configured to only accept queries from internal hosts (e.g., Stub Resolver host).

141 38. **Checklist Item 38:** Whenever Aggregate Caches are deployed, the forwarders must be
142 configured to be Validating Resolvers.

143 39. **Checklist item 39:** Each recursive server must have a root hints file containing the IP
144 address of one or more DNS root servers. The information in the root hints file should be
145 periodically checked for correctness.

146 40. **Checklist item 40:** The root hints file should be owned by the account under which the
147 name server software is run. The permission bits should be set so that the root hints file can
148 be read or modified only by the account that runs the name server software.

149 41. **Checklist item 41:** Administrators should configure two or more recursive resolvers for each
150 stub resolver on the network.

151 42. **Checklist item 42:** Enterprise firewalls should consider restricting outbound DNS traffic
152 from stub resolvers to only the enterprise's designated recursive resolvers.

153 43. **Checklist item 43:** Each recursive server must have a root hints file containing the IP
154 address of one or more DNS root servers. The information in the root hints file should be
155 periodically checked for correctness.

156 44. **Checklist item 44:** The root hints file should be owned by the account under which the
157 name server software is run. The permission bits should be set so that the root hints file can
158 be read or modified only by the account that runs the name server software.

159 45. **Checklist item 45:** Administrators should configure two or more recursive resolvers for each
160 stub resolver on the network.

161 46. **Checklist item 46:** Enterprise firewalls should consider restricting outbound DNS traffic
162 from stub resolvers to only the enterprise's designated recursive resolvers.

163 47. **Checklist item 47:** Non-validating stub resolvers (both DNSSEC-aware and non-DNSSEC-
164 aware) must have a trusted link with a validating recursive resolver.

165 48. **Checklist item 48:** Validators should routinely log any validation failures to aid in diagnosing
166 network errors.

167 49. **Checklist item 49:** Mobile or nomadic systems should either perform their own validation
168 or have a trusted channel back to a trusted validator.

169 50. **Checklist item 50:** Mobile or nomadic systems that perform its own validation should have
170 the same DNSSEC policy and trust anchors as validators on the enterprise network.

171 51. **Checklist item 51:** Validator administrator must configure one or more trust anchors for
172 each validator in the enterprise.

173 52. **Checklist item 52:** The validator administrator regularly checks each trust anchor to ensure
174 that it is still in use, and updates the trust anchor as necessary.

175

Appendix E Overview of Products Contributed by Collaborators

Components provided by collaborators included Mail User Agents (MUAs), Mail Transfer Agents (MTAs), and DNS Services. Most of the products included were DNS service components, but these DNS service components were initially provided with MUAs and MTAs in all cases. Where the MUA and MTA components employed are not part of the collaborator's standard offering, open source MUA and MTA components were included in the initial collaborator installation. Component overviews follow:

E.1 Open Source MUA and MTA Components

E.1.1 Thunderbird Mail User Agent

Mozilla Thunderbird is a free, open source, cross-platform email, news, and chat client developed by the Mozilla Foundation. Thunderbird is an email, newsgroup, news feed, and chat (XMPP, IRC, Twitter) client. The Mozilla Lightning extension, which is installed by default, adds PIM functionality. Thunderbird can manage multiple email, newsgroup, and news feed accounts and supports multiple identities within accounts. Features such as quick search, saved search folders (virtual folders), advanced message filtering, message grouping, and labels help manage and find messages. On Linux-based systems, system mail (movemail) accounts are supported. Thunderbird incorporates a Bayesian spam filter, a whitelist based on the included address book, and can also understand classifications by server-based filters such as SpamAssassin.

Thunderbird has native support for RFC 3851 S/MIME, but RFC 5757 (S/MIME version 3.2) is not supported. Support for other security systems can be added by installing extensions (e.g., the Enigmail extension adds support for PGP). S/MIME and PGP cannot both be used in the same message. SSL/TLS is also supported, but it is used only to temporarily encrypt data being sent and received between an email client and server. SSL/TLS can work in combination with S/MIME or OpenPGP.

Thunderbird supports POP and IMAP. It also supports LDAP address completion. Thunderbird supports the S/MIME standard, extensions such as Enigmail add support for the OpenPGP standard. A list of supported IMAP extensions can be found at wiki.mozilla.org. Since version 38, Thunderbird has integrated support for automatic linking of large files instead of attaching them directly to the mail message.

Thunderbird runs on a variety of platforms. Releases available on the primary distribution site support Linux, Windows, and OS X operating systems. Unofficial ports are available for FreeBSD, OpenBSD, OpenSolaris, OS/2, and eComStation.

E.1.2 Dovecot

Dovecot is used in the DNS-Based Email Security project to permit MUA access to the Postfix MTA. Dovecot is an open source IMAP¹ and POP3 email server for Linux/UNIX-like systems,

written with security primarily in mind. Dovecot is used in both small and large installations. It is compact and requires no special administration and it uses very little memory. Dovecot supports the standard mbox and Maildir formats. The mailboxes are transparently indexed and provide full compatibility with existing mailbox handling tools. Dovecot v1.1 passes all IMAP server standard compliance tests. Dovecot allows mailboxes and their indexes to be modified by multiple computers at the same time, providing compatibility with clustered file systems. Caching problems can be worked around with director proxies. Postfix 2.3+ and Exim 4.64+ users can do SMTP authentication directly against Dovecot's authentication backend without having to configure it separately, and Dovecot supports easy migration from many existing IMAP and POP3 servers, allowing the change to be transparent to existing users.

Dovecot currently offers IMAP4rev1, POP3, IPv6, SSL and TLS support. It supports multiple commonly used IMAP extensions, including SORT, THREAD and IDLE. Shared mailboxes are supported in v1.2+. Maildir++ quota is supported, but hard file system quota can introduce problems. Dovecot is commonly used with Linux, Solaris, FreeBSD, OpenBSD, NetBSD and Mac OS X. See the Dovecot Wiki page (<http://wiki2.dovecot.org/OSCompatibility>) about OS compatibility for more.

E.1.3 Postfix

Postfix is a free and open-source mail transfer agent (MTA) that routes and delivers electronic mail. Postfix is released under the IBM Public License 1.0 which is a free software license. As an SMTP client, Postfix implements a high-performance parallelized mail-delivery engine. Postfix is often combined with mailing-list software (such as Mailman).

Postfix consists of a combination of server programs that run in the background, and client programs that are invoked by user programs or by system administrators. The Postfix core consists of several dozen server programs that run in the background, each handling one specific aspect of email delivery. Examples are the SMTP server, the scheduler, the address rewriter, and the local delivery server. For damage-control purposes, most server programs run with fixed reduced privileges, and terminate voluntarily after processing a limited number of requests. To conserve system resources, most server programs terminate when they become idle.

Client programs run outside the Postfix core. They interact with Postfix server programs through mail delivery instructions in the user's `~/.forward` file, and through small "gate" programs to submit mail or to request queue status information.

As an SMTP server, Postfix implements a first layer of defense against spambots and malware. Administrators can combine Postfix with other software that provides spam/virus filtering (e.g., Amavisd-new), message-store access (e.g., Dovecot), or complex SMTP-level access-policies (e.g., postfwd, policyd-weight or greylisting).

1. The Internet Message Access Protocol (IMAP) is a mail protocol used for accessing email on a remote web server from a local client. IMAP and POP3 are the two most commonly used Internet mail protocols for retrieving emails. Both protocols are supported by all modern email clients and web servers.

74 Features include:

- 75 ■ standards-compliant support for SMTPUTF8, SMTP, LMTP, STARTTLS encryption including
DANE protocol support and “perfect” forward secrecy, SASL authentication, MIME
encapsulation and transformation, DSN delivery status notifications, IPv4, and IPv6
- 76 ■ configurable SMTP-level access policy that automatically adapts to overload
- 77 ■ virtual domains with distinct address-namespaces
- 78 ■ UNIX-system interfaces for command-line submission, for delivery to command, and for
direct delivery to message stores in mbox and maildir format
- 79 ■ light-weight content inspection based on regular expressions
- 80 ■ database lookup mechanisms including Berkeley DB, CDB, OpenLDAP LMDB, Memcached,
LDAP and multiple SQL database implementations
- 81 ■ a scheduler that implements parallel deliveries, with configurable concurrency and back-off
strategies
- 82 ■ a scalable zombie blocker that reduces SMTP server load due to botnet spam

83 Postfix extensions use the SMTP or Milter (Sendmail mail filter) protocols which both give full
84 control over the message envelope and content, or a simple text-based protocol that enables
85 complex SMTP-level access control policies. Extensions include:

- 86 ■ deep content inspection before or after a message is accepted into the mail queue
- 87 ■ mail authentication with DMARC, DKIM, SPF, or other protocols
- 88 ■ SMTP-level access policies such as greylisting or rate control

89 Postfix runs on BSD, GNU/Linux, OS X, Solaris and most other Unix-like operating system,
90 generally ships with a C compiler, and delivers a standard POSIX development environment. It is
91 the default MTA for the OS X, NetBSD and Ubuntu operating systems.

97 E.2 Microsoft Windows-Based Components

98 Microsoft's contribution includes a complete MUA, MTA, and DNS service stack, though each of
99 the components can be integrated into systems provided by other contributors.

100 E.2.1 Outlook

101 Microsoft Outlook is a personal information manager from Microsoft, available as a part of the
102 Microsoft Office suite. Although often used mainly as an email application, it also includes a
103 calendar, task manager, contact manager, note taking, journal, and web browsing. It can be
104 used as a stand-alone application, or can work with Microsoft Exchange Server and Microsoft
105 SharePoint Server for multiple users in an organization, such as shared mailboxes and
106 calendars, Exchange public folders, SharePoint lists, and meeting schedules. Microsoft has also
107 released mobile applications for most mobile platforms, including iOS and Android. Developers
108 can also create their own custom software that works with Outlook and Office components
109 using Microsoft Visual Studio. In addition, mobile devices can synchronize almost all Outlook
110 data to Outlook Mobile. Microsoft Outlook mail system uses the proprietary Messaging

111 Application Programming Interface (MAPI) to access Microsoft Exchange electronic mail
112 servers.

113 Outlook supports S/MIME (Secure/Multipurpose Internet Mail Extensions) is a standard for
114 public key encryption and signing of MIME data. S/MIME is on an IETF standards track and
115 defined in a number of documents, most importantly RFCs 3369, 3370, 3850 and 3851.

116 E.2.2 Exchange

117 Microsoft Exchange Server is a calendaring and mail server developed by Microsoft that runs
118 exclusively on the Microsoft Windows Server product line. Exchange Server was initially
119 Microsoft's internal mail server but is now published outside Microsoft. It uses the Active
120 Directory directory service. It is bundled with the Outlook email client.

121 Exchange Server supports POP3, IMAP, SMTP and EAS. It also supports IPv6, SMTP over TLS, PoP
122 over TLS, NNTP, and SSL. Exchange Server is licensed both in the forms of on-premises software
123 and software as a service. In the on-premises form, customer purchase client access licenses
124 (CALs). In the software as a service form, Microsoft receives a monthly service fee instead (see
125 https://en.wikipedia.org/wiki/Office_365).

126 E.2.3 Server DNS Services

127 Windows Server 2016 is a server operating system developed by Microsoft as part of the
128 Windows NT family of operating systems, developed concurrently with Windows 10. Microsoft
129 Server features server virtualization, networking, server management and automation, a web
130 and application platform, access and information protection, and virtual desktop infrastructure.
131 Key operating system elements for the DNS-Based Email Security project are Active Directory
132 and DNS Server.

133 E.2.3.1 Active Directory

134 Active Directory (AD) is a directory service that Microsoft developed for Windows domain
135 networks. It is included in most Windows Server operating systems as a set of processes and
136 services. Initially, Active Directory was only in charge of centralized domain management.
137 Active Directory is an umbrella title for a broad range of directory-based identity-related
138 services. A server running Active Directory Domain Services (AD DS) is called a domain
139 controller. It authenticates and authorizes all users and computers in a Windows domain type
140 network-assigning and enforcing security policies for all computers and installing or updating
141 software. For example, when a user logs into a computer that is part of a Windows domain,
142 Active Directory checks the submitted password and determines whether the user is a system
143 administrator or normal user.

144 Active Directory uses Lightweight Directory Access Protocol (LDAP) versions 2 and 3, Microsoft's
145 version of Kerberos, and DNS. Active Directory Domain Services (AD DS) is the cornerstone of
146 every Windows domain network. It stores information about members of the domain, including
147 devices and users, verifies their credentials and defines their access rights. The server (or the
148 cluster of servers) running this service is called a domain controller. A domain controller is
149 contacted when a user logs into a device, accesses another device across the network, or runs a
150 line-of-business Metro-style application side loaded into a device. Other Active Directory
151 services (excluding LDS, as well as most of Microsoft server technologies rely on or use Domain

152 Services; examples include Group Policy, Encrypting File System, BitLocker, Domain Name
153 Services, Remote Desktop Services, Exchange Server and SharePoint Server.

154 Active Directory Certificate Services (AD CS) establishes an on-premises public key
155 infrastructure. It can create, validate and revoke public key certificates for internal uses of an
156 organization. These certificates can be used to encrypt files (when used with Encrypting File
157 System), emails (per S/MIME standard), network traffic (when used by virtual private networks,
158 Transport Layer Security protocol or IPSec protocol).

159 E.2.3.2 DNS Server

160 Microsoft Windows server operating systems can run the DNS Server service, a monolithic DNS
161 server that provides many types of DNS service, including caching, Dynamic DNS update, zone
162 transfer, and DNS notification. DNS notification implements a push mechanism for notifying a
163 select set of secondary servers for a zone when it is updated. DNS Server has improved
164 interoperability with BIND and other implementations in terms of zone file format, zone
165 transfer, and other DNS protocol details.

166 Microsoft's DNS server supports different database back ends. Microsoft's DNS server supports
167 two such back ends. DNS data can be stored either in master files (also known as zone files) or
168 in the Active Directory database itself. In the latter case, since Active Directory (rather than the
169 DNS server) handles the actual replication of the database across multiple machines, the
170 database can be modified on any server (multiple-master replication), and the addition or
171 removal of a zone will be immediately propagated to all other DNS servers within the
172 appropriate Active Directory "replication scope". (Contrast this with BIND, where when such
173 changes are made, the list of zones, in the /etc/named.conf file, has to be explicitly updated on
174 each individual server.)

175 Microsoft's DNS server can be administered using either a graphical user interface, the DNS
176 Management Console, or a command line interface, the dnscmd utility. New to Windows
177 Server 2012 is a fully featured PowerShell provider for DNS server management.

178 E.3 NLnet Labs Name Server Daemon-Based 179 Components

180 E.3.1 NSD4 Authoritative Name Server

181 Name Server Daemon (NSD) is an open-source DNS server. It was developed from scratch by
182 NLnet Labs of Amsterdam in cooperation with the RIPE NCC, as an authoritative name server
183 (i.e., not implementing the recursive caching function by design). The intention of this
184 development is to add variance to the "gene pool" of DNS implementations originally intended
185 for root servers, top-level domains (TLDs) and second-level domains (SLDs), thus increasing the
186 resilience of DNS against software flaws or exploits.

187 NSD uses BIND-style zone-files (zone-files used under BIND can usually be used unmodified in
188 NSD, once entered into the NSD configuration).

189 The collection of programs/processes that make-up NSD are designed so that the NSD daemon
190 itself runs as a non-privileged user and can be easily configured to run in a Chroot jail, such that

191 security flaws in the NSD daemon are not so likely to result in system-wide compromise as
 192 without such measures.

193 The latest current stable release is NSD 4.1.13. Download the latest version here: <https://www.nlnetlabs.nl/downloads/nsd/nsd-4.1.10.tar.gz>.

194 NSD is thoroughly tested, there is a regression tests report available.

195 For NSD 4, the memory estimation tool can be compiled in the source tarball with `make nsd-mem` and running it on a config file with the zone files in question.

196 NLnet Labs has a long-term commitment for supporting NSD. There will be an advanced notice
 197 when the organization's commitment ends. The latest NSD release will supported for at least
 200 two years after an end-of-life notification has been sent to the community.

201 Manual pages are installed, they can also be viewed:

- 202 1. nsd(8) man page: <https://www.nlnetlabs.nl/projects/nsd/nsd.8.html>
- 203 2. nsd-control(8) man page: <https://www.nlnetlabs.nl/projects/nsd/nsd-control.8.html>
- 204 3. nsd-checkconf(8) man page: <https://www.nlnetlabs.nl/projects/nsd/nsd-checkconf.8.html>
- 205 4. nsd-checkzone(8) man page: <https://www.nlnetlabs.nl/projects/nsd/nsd-checkzone.8.html>
- 206 5. nsd.conf(5) man page: <https://www.nlnetlabs.nl/projects/nsd/nsd.conf.5.html>

207 NSD users can subscribe to nsd-users and browse the archives of nsd-users here <http://open.nlnetlabs.nl/mailman/listinfo/nsd-users/>.

208 The repository of NSD is available at `/svn/nsd/`, the NSD 4.x.x development tree is located in
 210 `trunk/`.

211 E.3.2 OpenDNSSEC Domain Name Security Manager

212 OpenDNSSEC software manages the security of domain names on the Internet. The
 213 OpenDNSSEC project is a cooperative effort intended to drive adoption of Domain Name
 214 System Security Extensions (DNSSEC) in order to further enhance Internet security.

215 OpenDNSSEC was created as an open-source turn-key solution for DNSSEC. It secures DNS zone
 216 data just before it is published in an authoritative name server. OpenDNSSEC takes in unsigned
 217 zones, adds digital signatures and other records for DNSSEC and passes it on to the
 218 authoritative name servers for that zone. OpenDNSSEC will furthermore take care of the key
 219 management and roll-over procedure to replace keys. It acts as a bump in the wire, where it
 220 will fit in an existing DNS tool chain without modification in that tool chain. Incrementally
 221 incorporating changes and re-using already signed zones to perform a constant up-to-date
 222 zone.

223 All keys are stored in a hardware security module and accessed via PKCS #11, a standard
 224 software interface for communicating with devices which hold cryptographic information and
 225 perform cryptographic functions. OpenDNSSEC uses SoftHSM, OpenSSL, the Botan
 226 cryptographic library, and SQLite or MySQL as database back-end. It is used on the .se, .dk, .nl
 227 .ca, .za, .uk, and other top-level domains. OpenDNSSEC can be downloaded from:

- 228 ■ <https://dist.opendnssec.org/source/opendnssec-2.0.1.tar.gz>
- 229 ■ <https://dist.opendnssec.org/source/opendnssec-2.0.1.tar.gz.sig>

230 ■ Checksum SHA256:
 231 bf874bbb346699a5b539699f90a54e0c15fff0574df7a3c118abb30938b7b346

232 In August of 2014, NLnet Labs took responsibility for continuing the OpenDNSSEC activities of
 233 both the OpenDNSSEC software project and the Swedish OpenDNSSEC AB.

234 E.3.3 Unbound DNS Resolver

235 Unbound is a validating, recursive, and caching DNS resolver. The C implementation of
 236 Unbound is developed and maintained by NLnet Labs. It is based on ideas and algorithms taken
 237 from a Java prototype developed by Verisign labs, Nominet, Kirei and ep.net. Unbound is
 238 designed as a set of modular components, so that also DNSSEC (secure DNS) validation and
 239 stub-resolvers (that do not run as a server, but are linked into an application) are easily possible.

240 The source code is under a BSD License.

241 Release 1.5.9 of Unbound was released June 9, 2016. The repository for unbound is available
 242 <https://unbound.nlnetlabs.nl/svn/>. The development tree is located in trunk/.

243 The latest source code tarball is available for download.

244 Unbound problems can be reported through the NLnet Labs bugzilla web interface. In the case
 245 NLnet Labs will stop supporting the product, and they will announce such two years in advance.
 246 Unbound is subject to NLnet Labs Security Patch Policy. Commercial support for Unbound is
 247 available from several organizations.

248 E.4 ISC BIND Component

249 Internet Systems Consortium, Inc., also known as ISC, is a non-profit corporation that supports
 250 the infrastructure of the Internet by developing and maintaining core production-quality
 251 software, protocols, and operations. ISC has developed several key Internet technologies that
 252 enable the global Internet, including BIND.

253 BIND is open source software that implements the Domain Name System (DNS) protocols for
 254 the Internet. It is a reference implementation of those protocols, but it is also production-grade
 255 software, suitable for use in high-volume and high-reliability applications. The acronym BIND
 256 stands for Berkeley Internet Name Domain, because the software originated in the early 1980s
 257 at the University of California at Berkeley.

258 BIND is widely used DNS software that provides a stable platform on top of which organizations
 259 can build distributed computing systems that are fully compliant with published DNS standards.

260 BIND is transparent open source. If an organization needs some functionality that is not in
 261 BIND, it is possible to modify it, and contribute the new feature back to the community by
 262 sending ISC its source. It is possible to download a tar ball from the ISC web site (<https://www.isc.org/downloads/>), [ftp.isc.org \(http://ftp.isc.org/isc/bind9/cur/\)](http://ftp.isc.org/isc/bind9/cur/), or a binary from an
 264 organization's operating system repository.

265 The BIND software distribution has three parts:

266 E.4.1 Domain Name Resolver

267 The BIND resolver is a program that resolves questions about names by sending those
268 questions to appropriate servers and responding appropriately to the servers' replies. In the
269 most common application, a web browser uses a local stub resolver library on the same
270 computer to look up names in the DNS. That stub resolver is part of the operating system.
271 (Many operating system distributions use the BIND resolver library.) The stub resolver usually
272 will forward queries to a caching resolver, a server or group of servers on the network
273 dedicated to DNS services. Those resolvers will send queries to one or multiple authoritative
274 servers in order to find the IP address for that DNS name.

275 DNS authoritative operations include the following features:

- 276 **1. NXDOMAIN Redirect:** When a user searches for a non-existent domain, (NXDOMAIN
277 response) the user can be redirected to another web page. This is done using the BIND DLZ
278 feature.
- 279 **2. Flexible Cache Controls:** From time to time users can get incorrect or outdated records in
280 the resolver cache. BIND gives users the ability to remove them selectively or wholesale.
- 281 **3. Split DNS:** BIND provides the ability to configure different views in a single BIND server. This
282 allows users to give internal (on-network) and external (from the Internet) users different
283 views of DNS data, keeping some DNS information private.
- 284 **4. Cache Hit Rate Optimization:** BIND is designed to be persistent and resilient in resolving
285 queries even when there is a delay in responding, in order to populate the cache for later
286 requests. DNS Pre-fetch is a technique for continuously refreshing the cached records for
287 popular domains, reducing the time the user has to wait for a response.
- 288 **5. Resolver Rate-limiting:** Beginning with BIND 9.10.3, two new configuration parameters
289 were added, *fetches-per-zone* and *fetches-per-server*. These features enable rate-limiting
290 queries to authoritative systems that appear to be under attack. These features have been
291 successful in mitigating the impact of a DDOS attack on resolvers in the path of the attack.
- 292 **6. DNSSEC Validation:** DNSSEC validation protects clients from impostor sites. In BIND, this is
293 enabled with a single command. BIND supports RFC 5011 maintenance of root key trust
294 anchors. BIND also has a Negative Trust Anchor feature (introduced in the 9.9 subscription
295 branch), which temporarily disables DNSSEC validation when there is a problem with the
296 authoritative server's DNSSEC support.
- 297 **7. Geo IP:** Geolocated IP, or Geographic IP, allows a BIND DNS server to provide different responses
298 based on the network information about the recursive DNS resolver that a user is using.
299 There is an active Internet Draft describing another mechanism for providing location
300 information, called EDNS-Client-Subnet-Identifier. This requires the resolver to cache
301 multiple different addresses for a given DNS record, depending on the address of the
302 requester. This feature has not been added to the BIND9 resolver, although the
303 corresponding feature has been developed for the BIND9 authoritative server.
- 304 **8. Response Policy Zone:** A Response Policy Zone or RPZ is a specially-constructed zone that
305 specifies a policy rule set. The primary application is for blocking access to zones that are
306 believed to be published for abusive or illegal purposes. There are companies who
307 specialize in identifying abuse sites on the Internet, who market these lists in the form of
308 RPZ feeds. For more information on RPZ, including a list of DNS reputation feed providers,
309 see <https://dnsrpz.info>.

310 E.4.2 Domain Name Authority Server

311 The authoritative DNS server answers requests from resolvers, using information about the
 312 domain names it is authoritative for. Enterprises can provide DNS services on the Internet by
 313 installing this software on a server and giving it information about the enterprise's domain
 314 names.

- 315 1. **Response Rate Limiting:** An enhancement to the DNS protocol to reduce the problem of
 316 "amplification attacks" by limiting DNS responses. Response rate limiting is on by default.
- 317 2. **Dynamically-Loadable Zones:** enable BIND9 to retrieve zone data directly from an external
 318 database. This is not recommended for high-query rate authoritative environments.
- 319 3. **Reload Time Reduction:** BIND server zone files can be updated via nsupdate, and 'dynamic'
 320 zone files can be added via RNDC, both without restarting BIND. For those times when it is
 321 necessary to restart, the **MAP** zone file format can speed up re-loading a large zone file into
 322 BIND, such as on restart.
- 323 4. **Hardware Security Modules:** BIND supports the use of Hardware Security Modules through
 324 either a native PKCS#11 interface, or the OpenSSL PKCS#11 provider. HSMs are used to
 325 store key material outside of BIND for security reasons.
- 326 5. **DNSSEC With In-line Signing:** BIND fully supports DNSSEC With In-line Signing and has an
 327 easy-to use implementation. Once an enterprise has initially signed its zones, BIND can
 328 automatically re-sign the records as they are updated with in-line signing, maintaining the
 329 DNSSEC validity of the records. BIND supports both NSEC and NSEC3 and inline signing
 330 works with NSEC3.
- 331 6. **Catalog Zones:** Catalog Zones were introduced in BIND 9.11.0 to facilitate the provisioning
 332 of zone information across a nameserver constellation. Catalog Zones are particularly useful
 333 when there are a large number of secondary servers. A special zone of a new type, a catalog
 334 zone, is set up on the master. Once a catalog zone is configured, when an operator wishes to
 335 add a new zone to the nameserver constellation s/he can provision the zone in one place
 336 only, on the master server and add an entry describing the zone to the catalog zone. As the
 337 secondary servers receive the updated copy of the catalog zone data they will note the new
 338 entry and automatically create a zone for it. Deletion of a zone listed in a catalog zone is
 339 done by deleting the entry in the catalog zone on the master.
- 340 7. **Scalable Master/Slave Hierarchy:** A DNS authoritative system is composed of a zone
 341 primary or master with one or more slave servers. Zones files are established and updated
 342 on a master BIND server. Slaves maintain copies of the zone files and answer queries. This
 343 configuration allows scaling the answer capacity by adding more slaves, while zone
 344 information is maintained in only one place. The master signals that updated information is
 345 available with a notify message to the slaves, and the slaves then initiate an update from
 346 the master. BIND fully supports both the AXFR (complete transfer) and IXFR (incremental
 347 transfer) methods, using the standard TSIG security mechanism between servers. There are
 348 a number of configuration options for controlling the zone updating process.

349 E.4.3 Tools

350 ISC includes a number of diagnostic and operational tools. Some of them, such as the popular
 351 DIG tool, are not specific to BIND and can be used with any DNS server.

352 E.5 Secure64 Component

353 The Secure64 contributions included an automated online Secure64 DNS Signer delivered on
 354 dedicated hardware and DNSSEC-capable VM images of DNS Cache, DNS Authority, and DNS
 355 Manager. DNS Manager provided centralized management of Secure64 DNS Cache software
 356 and configurations and provided network-wide monitoring of key performance indicators. DNS
 357 Manager allowed creation of groups of servers and assignment of configurations to a group, a
 358 single server, or all servers. DNS Authority is an authoritative signer and server as a single
 359 platform. This stack was able to demonstrate Outlook, Thunderbird, or Apple Mail as MUAs and
 360 uses Postfix as an MTA and Dovecot to provide IMAP for clients. Descriptions of the DNS service
 361 components follow:

362 E.5.1 DNS Signer

363 Secure64 DNS Signer is DNSSEC key management and zone signing software that is designed to
 364 facilitate and provide security for DNSSEC implementation. Secure64 DNS Signer fully
 365 automates DNSSEC key generation, key rollover, zone signing and re-signing processes. It is
 366 designed to scale to large, dynamic environments by maintaining DNSSEC signing keys securely
 367 online while providing incremental zone signing and high signing performance. Signer
 368 integrates into existing infrastructures configurations. It is fully compatible with Secure64 DNS
 369 Authority, BIND, NSD, and Microsoft DNS masters and slaves. Signer supports all of the RFCs
 370 and best practices required to deploy DNSSEC.

371 E.5.2 DNS Authority

372 Secure64 DNS Authority is a name server software product. It provides built-in DoS protection
 373 that identifies and blocks TCP or UDP attack traffic. It is designed to respond to legitimate
 374 queries, even while under attack. DNS Authority provides real-time alerts and attack
 375 characteristics through syslog and SNMP traps in order to enable remedial action. Authority is
 376 also designed to be anycasted in any data center, even for enterprises that don't operate the
 377 routing infrastructure. The administrator can insert and withdraw servers without requiring
 378 router changes or deploying dedicated router hardware. Authority directly reads existing BIND
 379 configuration files and is interoperable with name servers running BIND, NSD, or Microsoft
 380 Windows DNS software. Some specific features include the following:

- 381 **1. IPv6 support:** Authority supports IPv6 in either dual stack or IPv6-only mode.
- 382 **2. PipeProtector:** Authority's PipeProtector™ feature protects networks by automatically
 383 identifying the sources of amplified flood attacks and communicating with the upstream
 384 router to blackhole the attack traffic.
- 385 **3. Built-in BGP:** Built-in Border Gateway Protocol (BGP) permits Authority to be set up in an
 386 anycast configuration, which provides greater resiliency against denial-of-service attacks
 387 and improved performance. After BGP is initially configured, the administrator can insert
 388 and withdraw the server from the anycast cluster without making router changes.
- 389 **4. Secured runtime environment:** Authority is designed to run on a SourceT operating system
 390 and to utilize server hardware security capabilities to eliminate all paths for injection or
 391 execution of malicious code at runtime.

392 5. **System Authentication:** Digital signatures of the firmware, operating system and
 393 application code are all validated during the boot process. This protects against the
 394 operating system and the application code images on disk from being compromised by a
 395 rootkit.

396 6. **Secured zone data:** Authority provides end-to-end integrity protection of zone data by
 397 supporting DNSSEC, TSIG and ACLs for queries, notifies and zone transfers.

398 7. **Synthesized PTR records:** Reverse DNS records for IPv6 addresses or other large address
 399 blocks can be generated on the fly where necessary to preserve compatibility with other
 400 systems that rely upon the existence of these reverse records.

401 8. **Standards support:** Authority supports ENUM standards, including RFC 3163 (SIP initiation
 402 protocol), RFC 6116 (storage of data for E.164 numbers in the DNS) and 3GPP TS 29.303
 403 (DNS procedures for the Evolved Packet System).

404 9. **Split horizon DNS:** Views permit configuration of an authoritative server to provide
 405 different functionality and responses based on characteristics of the requesting client.

406 E.5.3 DNS Cache

407 Secure64 DNS Cache is scalable, secure, caching DNS software designed to provide built-in
 408 protection against high volume denial-of-service attacks and immunity to BIND-specific security
 409 vulnerabilities. DNS Guard is a family of security services that protect users and the network
 410 from malicious activity, while the Web Error Redirection Module allows service providers to
 411 improve the end user's experience while generating incremental revenues that flow right to the
 412 bottom line. Some specific features include the following:

413 1. **IPv6 Support:** DNS Cache supports both dual stack and deployment of a pure IPv6 network
 414 while providing compatibility with IPv4 networks.

415 2. **Built-In DDoS Protection:** Built-in DDoS detection and mitigation allows DNS Cache to
 416 continue to respond to legitimate queries while fending off high volume denial-of-service-
 417 attacks. This combats a common issue with DNS solutions that crash or become unavailable
 418 at lower levels of attack traffic. In addition to mitigating high volume attacks, DNS Cache
 419 automatically detects cases of individual clients exceeding a user-defined query threshold
 420 and temporarily blacklists them while logging information about the offending client. This
 421 helps prevent inadvertent participation in a denial-of-service attack.

422 3. **SNMP:** DNS Cache provides several MIBs, that allow monitoring of the chassis, network,
 423 operating system and application in real time and support a variety of network monitoring
 424 systems. In addition, DNS Cache directly provides alerts of critical operational conditions
 425 through SNMP traps without requiring special configuration within the network monitoring
 426 system.

427 4. **Centralized management:** DNS Cache servers can be managed individually, or can be
 428 centrally managed and monitored through Secure64 DNS Manager.

429 5. **Scalable performance:** At a 90% cache hit rate, DNS Cache delivers over 125,000 queries
 430 per second, which can easily be increased to 280,000 queries per second through the
 431 optional software-based Capacity Expansion Module.

432 6. **DNSSEC validation overrides:** DNS Cache can configured to validate DNSSEC signed
 433 answers. Because DNSSEC configuration errors are not uncommon, operators can readily

434 identify domains failing validation and specify which of these should be allowed to resolve
435 normally.

436 7. **Merge Zones:** DNS Cache's merge zones feature allows a number of dynamic authoritative
437 zones to be split up among different authoritative servers, each of which is queried for a
438 response to a query for that zone until an answer is received.

439 8. **Web Error Redirection Module:** The optional Web Error Redirection Module allows service
440 providers to redirect NXDOMAIN responses from authoritative servers to a provider-
441 branded search portal that helps guide users to their intended designation.

442 9. **Rules engine:** DNS Cache's rules engine provides fine-grained control over which responses
443 are redirected, and includes built-in support for opt-out.

444 E.5.4 DNS Manager

445 DNS Manager provides centralized management of Secure64 DNS Cache software and
446 configurations and provides network-wide monitoring of key performance indicators. This GUI
447 based application can configure, manage, and monitor a set of Secure64 DNS Cache servers
448 from one central point. In an environment consisting of many DNS servers, there are likely to be
449 differences in configurations. Some servers may be anycasted, while others are load balanced,
450 for example. Or servers located in different geographies may have different values for local DNS
451 data. DNS Manager allows creation of groups of servers and assigns configurations to a group, a
452 single server, or all servers. Groups may be arranged hierarchically. Common configuration
453 parameters may be assigned to all servers in the network, whereas settings specific to subsets
454 of servers may be assigned at the group level, and IP addresses and other server-specific
455 information are assigned to each specific server. All actions to modify configuration files or
456 software versions are revision controlled and logged. Authorized users can rollback to previous
457 software versions or configurations if necessary. DNS Manager is able to monitor key
458 performance indicators across the DNS network, including queries per second, CPU, disk and
459 memory utilization.

460 E.5.5 Secure64 Apple Key Chain Utility

461 The Apple Key Chain Utility is a Secure64 utility for Public Key Retrieval into the Apple Key
462 Chain. This utility is delivered on a MacBook loaded with Apple Mail and is a program for the
463 MacBook that will fetch SMIMEA records and put them in the keystore so that we can
464 demonstrate end-to-end security.

Appendix F Installation and Configuration Log for NSD4, Unbound, and OpenDNSSEC

4 The following log captures the installation and configuration process for NSD4, Unbound, and
5 OpenDNSSEC for the NCCoE's DNS-Based Email Security project. Please note that the IP
6 addresses, domain names, and mail addresses are for the NCCoE laboratory and must not be
7 used in actual implementations.

```
8 #####
9 # Unbound installation log for 10.33.XX.XX
10 #####
11
12
13 # Unbound does not depend on a resolver for its installation. However, I
14 # configure one here so I can use yum from installation of the dependencies.
15 [rdolmans@unbound ~]$ sudo cp /etc/resolv.conf /etc/resolv.conf.orig
16 [rdolmans@unbound ~]$ echo "nameserver 10.97.XX.X" | sudo tee -a /etc/resolv.conf
17
18
19 # Install build tools
20 [rdolmans@unbound ~]$ sudo yum group install "Development Tools"
21
22
23 # Install unbound dependencies: openssl, expat
24 [rdolmans@unbound ~]$ sudo yum install openssl-devel expat-devel
25
26
27 # Download Unbound and verify
28 [rdolmans@unbound ~]$ curl https://unbound.net/downloads/unbound-1.5.8.tar.gz -o unbound-
29 1.5.8.tar.gz
30 [rdolmans@unbound ~]$ cat unbound-1.5.8.tar.gz | openssl sha256
31 (stdin)= 33567a20f73e288f8daa4ec021fbb30fe1824b346b34f12677ad77899ecd09be
32
33
34 # We do not need a nameserver anymore, move back old resolv.conf
35 [rdolmans@unbound ~]$ sudo mv /etc/resolv.conf.orig /etc/resolv.conf
36
37
38 # extract, ./configure, compile and install Unbound
39 [rdolmans@unbound ~]$ tar xvzf unbound-1.5.8.tar.gz
40 [rdolmans@unbound ~]$ cd unbound-1.5.8
41 [rdolmans@unbound unbound-1.5.8]$ ./configure
42 [rdolmans@unbound unbound-1.5.8]$ make
43 [rdolmans@unbound unbound-1.5.8]$ sudo make install
44
45
46 # Add system user and group
```

```

47 [rdolmans@unbound unbound-1.5.8]$ sudo groupadd -r unbound
48 [rdolmans@unbound unbound-1.5.8]$ sudo useradd -r -g unbound -s /sbin/nologin -c "unbound name
49 daemon" unbound
50
51
52 # Setup unbound-control, get trust anchor
53 [rdolmans@unbound ~]$ sudo unbound-control-setup
54 [rdolmans@unbound ~]$ sudo unbound-anchor
55
56
57 # Config changes:
58 # 1. Specify the interfaces to listen on
59 # 2. Allow second host to use this resolver (ACL)
60 # 3. Load DNSSEC trust anchor obtained using unbound-anchor
61 # 4. Enable remote-control (for unbound-control command, limited to localhost)
62
63
64 [rdolmans@unbound ~]$ diff -u /usr/local/etc/unbound/unbound.conf.orig /usr/local/etc/unbound/
65 unbound.conf
66 --- /usr/local/etc/unbound/unbound.conf.orig      2016-05-10 09:22:13.917495389 -0400
67 +++ /usr/local/etc/unbound/unbound.conf 2016-05-12 06:34:02.660574284 -0400
68 @@ -34,6 +34,9 @@
69         # specify 0.0.0.0 and ::0 to bind to all available interfaces.
70         # specify every interface[@port] on a new 'interface:' labelled line.
71         # The listen interfaces are not changed on reload, only on restart.
72 +        interface: 192.168.3.98
73 +        interface: ::1
74 +        interface: 127.0.0.1
75         # interface: 192.0.2.153
76         # interface: 192.0.2.154
77         # interface: 192.0.2.154@5003
78 @@ -197,6 +200,7 @@
79         # access-control: ::0/0 refuse
80         # access-control: ::1 allow
81         # access-control: ::ffff:127.0.0.1 allow
82 +        access-control: 192.168.3.0/23 allow
83
84
85         # if given, a chroot(2) is done to the given directory.
86         # i.e. you can chroot to the working directory, for example,
87 @@ -376,7 +380,7 @@
88         # you start unbound (i.e. in the system boot scripts). And enable:
89         # Please note usage of unbound-anchor root anchor is at your own risk
90         # and under the terms of our LICENSE (see that file in the source).
91 -        # auto-trust-anchor-file: "/usr/local/etc/unbound/root.key"
92 +        auto-trust-anchor-file: "/usr/local/etc/unbound/root.key"
93
94
95         # File with DLV trusted keys. Same format as trust-anchor-file.

```

```

97      # There can be only one DLV configured, it is trusted from root down.
98 @@ -614,7 +618,7 @@
99 remote-control:
100      # Enable remote control with unbound-control(8) here.
101      # set up the keys and certificates with unbound-control-setup.
102 -      # control-enable: no
103 +      control-enable: yes
104
105      # Set to no and use an absolute path as control-interface to use
106      # a unix local named pipe for unbound-control.
107
108
109 # Start daemon
110 [rdolmans@unbound ~]$ sudo unbound-control start
111
112
113 # add local resolver to resolv.conf
114 [rdolmans@unbound ~]$ echo "nameserver ::1" | sudo tee -a /etc/resolv.conf
115
116 # Install ldns tools (incl. drill)
117 [rdolmans@unbound ~]$ sudo yum install ldns
118
119
120 # Test DNSSEC validation
121 # 1. resolve bogus record with CD bit set, should result in answer
122 # 2. resolve bogus record with CD bit unset, should result in SERVFAIL
123
124 # CD set:
125 [rdolmans@unbound ~]$ drill txt bogus.nlnetlabs.nl @::1 -o CD
126 ; ; ->>HEADER<<- opcode: QUERY, rcode: NOERROR, id: 36453
127 ; ; flags: qr rd cd ra ; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 2
128 ; ; QUESTION SECTION:
129 ; ; bogus.nlnetlabs.nl. IN      TXT
130
131
132 ; ; ANSWER SECTION:
133 bogus.nlnetlabs.nl.      59      IN      TXT      "will be Bogus"
134
135
136 ; ; AUTHORITY SECTION:
137 nlnetlabs.nl.      10200    IN      NS      sec2.authdns.ripe.net.
138 nlnetlabs.nl.      10200    IN      NS      anyns.pch.net.
139 nlnetlabs.nl.      10200    IN      NS      ns.nlnetlabs.nl.
140 nlnetlabs.nl.      10200    IN      NS      ns-ext1.sidn.nl.
141
142 ; ; ADDITIONAL SECTION:
143 ns.nlnetlabs.nl.      9831     IN      A       185.49.140.60
144 ns.nlnetlabs.nl.      9831     IN      AAAA    2a04:b900::8:0:0:60
145
146

```

```

147 ;; Query time: 581 msec
148 ;; SERVER: ::1
149 ;; WHEN: Thu May 12 05:58:20 2016
150 ;; MSG SIZE  rcvd: 209
151
152
153 # CD unset:
154 [rdolmans@unbound ~]$ drill txt bogus.nlnetlabs.nl @::1
155 ;; ->>HEADER<<- opcode: QUERY, rcode: SERVFAIL, id: 14388
156 ;; flags: qr rd ra ; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0
157 ;; QUESTION SECTION:
158 ;; bogus.nlnetlabs.nl.  IN      TXT
159
160 ;; ANSWER SECTION:
161
162 ;; AUTHORITY SECTION:
163
164 ;; ADDITIONAL SECTION:
165
166 ;; Query time: 0 msec
167 ;; SERVER: ::1
168 ;; WHEN: Thu May 12 05:59:06 2016
169 ;; MSG SIZE  rcvd: 36
170
171
172
173 ######
174 # NSD installation log for 10.33.XX.XX
175 ######
176
177 # Add 192.168.3.98 to resolv.conf
178 [rdolmans@nsd ~]$ echo "nameserver 192.168.3.98" | sudo tee -a /etc/resolv.conf
179
180 # install openssl, libevent
181 [rdolmans@nsd ~]$ sudo yum install openssl-devel libevent-devel
182
183 # SoftHSM
184 [rdolmans@nsd ~]$ tar xvzf softhsm-2.1.0.tar.gz
185 [rdolmans@nsd ~]$ cat softhsm-2.1.0.tar.gz | openssl sha256
186 (stdin)= 0399b06f196fbfaebe73b4aeff2e2d65d0dc1901161513d0d6a94f031dcd827e
187 [rdolmans@nsd softhsm-2.1.0]$ cd softhsm-2.1.0
188 [rdolmans@nsd softhsm-2.1.0]$ autoreconf -i -f
189 # openssl version has no gost support, disable
190 [rdolmans@nsd softhsm-2.1.0]$ ./configure --disable-gost
191 [rdolmans@nsd softhsm-2.1.0]$ make
192 [rdolmans@nsd softhsm-2.1.0]$ sudo make install
193 [rdolmans@nsd softhsm-2.1.0]$ sudo softhsm2-util--init-token--slot0--label"OpenDNSSEC"
194
195 # LDNS (incl. examples and drill)

```

```

196 [rdolmans@nsd ~]$ curl https://nlnetlabs.nl/downloads/ldns/ldns-1.6.17.tar.gz -o ldns-
197 1.6.17.tar.gz
198 [rdolmans@nsd ~]$ cat ldns-1.6.17.tar.gz | openssl sha1
199 (stdin)= 4218897b3c002aadfc7280b3f40cda829e05c9a4
200 [rdolmans@nsd ~]$ tar xvzf ldns-1.6.17.tar.gz
201 [rdolmans@nsd ~]$ cd ldns-1.6.17
202 [rdolmans@nsd ldns-1.6.17]$ ./configure --with-examples --with-drill
203 [rdolmans@nsd ldns-1.6.17]$ make
204 [rdolmans@nsd ldns-1.6.17]$ sudo make install
205
206 # OpenDNSSEC
207 # install dependencies: SQLite3, libxml2, java (for now)
208 [rdolmans@nsd ~]$ sudo yum install libxml2-devel sqlite-devel java-1.8.0-openjdk-devel
209 [rdolmans@nsd ~]$ git clone https://github.com/opendnssec/opendnssec.git
210 [rdolmans@nsd ~]$ cd opendnssec
211 [rdolmans@nsd opendnssec]$ sh autogen.sh
212 [rdolmans@nsd opendnssec]$ ./configure
213 [rdolmans@nsd opendnssec]$ make
214 [rdolmans@nsd opendnssec]$ sudo make install
215
216
217 # Setup SQLite db
218 [rdolmans@nsd opendnssec]$ sudo ods-enforcer-db-setup
219
220 # Use SoftHSM2, reload NSD zone after signing
221 [rdolmans@nsd ~]$ sudo diff -u /etc/opendnssec/conf.xml.sample /etc/opendnssec/conf.xml
222 --- /etc/opendnssec/conf.xml.sample      2016-05-12 10:53:35.154584441 -0400
223 +++ /etc/opendnssec/conf.xml      2016-05-17 12:03:20.719795941 -0400
224 @@ -5,9 +5,9 @@
225         <RepositoryList>
226
227             <Repository name="SoftHSM">
228                 <Module>/usr/local/lib/softhsm/libsofthsm.so</Module>
229                 <Module>/usr/local/lib/softhsm/libsofthsm2.so</Module>
230                 <TokenLabel>OpenDNSSEC</TokenLabel>
231                 <PIN>1234</PIN>
232                 <PIN>*****</PIN>
233                 <SkipPublicKey/>
234             </Repository>
235
236 @@ -87,9 +87,7 @@
237     <!--
238             <NotifyCommand>/usr/local/bin/my_nameserver_reload_command</
239 NotifyCommand>
240     -->
241     <!--
242             <NotifyCommand>/usr/sbin/rndc reload %zone</NotifyCommand>
243     -->
244     <NotifyCommand>/usr/local/sbin/nsd-control reload %zone</NotifyCommand>
245     </Signer>

```

```

246
247
248  </Configuration>
249
250
251 # Add policy to KASP config file. We use a policy named dnslab here, which is based on policy
252 default (but uses NSEC).
253 # See /etc/opendnssec/kasp.xml
254
255 [rdolmans@nsd ~]$ sudo ods-enforcer update all
256 Created policy dnslab successfully
257 Policy dnslab already up-to-date
258 update all completed in 0 seconds.
259 [rdolmans@nsd ~]$ sudo ods-enforcer policy list
260 Policy:                               Description:
261 dnslab                                Policy used for the NCCOE dnslab
262 policy list completed in 0 seconds.
263 [rdolmans@nsd ~]$ sudo ods-enforcer zone add --zone nev1.dnslab.nccoe.nist.gov --policy dnslab
264 Zone nev1.dnslab.nccoe.nist.gov added successfully
265 zone add completed in 1 seconds.
266
267
268
269 # NSD
270 # Download, verify checksum, extract, configure, compile and install NSD
271 [rdolmans@nsd ~]$ curl https://nlnetlabs.nl/downloads/nsd/nsd-4.1.9.tar.gz -o nsd-4.1.9.tar.gz
272 [rdolmans@nsd ~]$ cat nsd-4.1.9.tar.gz | openssl sha256
273 (stdin)= b811224d635331de741f1723aefc41adda0a0a3a499ec310aa01dd3b4b95c8f2
274 [rdolmans@nsd ~]$ tar xvzf nsd-4.1.9.tar.gz
275 [rdolmans@nsd ~]$ cd nsd-4.1.9
276 [rdolmans@nsd nsd-4.1.9]# ./configure --with-pidfile=/var/run/nsd/nsd.pid
277 [rdolmans@nsd nsd-4.1.9]$ make
278 [rdolmans@nsd nsd-4.1.9]$ sudo make install
279 [rdolmans@nsd ~]$ sudo nsd-control-setup
280
281 # enable in config
282 [rdolmans@nsd ~]$ sudo cp /etc/nsd/nsd.conf.sample /etc/nsd/nsd.conf
283 [rdolmans@nsd ~]$ diff -u /etc/nsd/nsd.conf.sample /etc/nsd/nsd.conf
284 --- /etc/nsd/nsd.conf.sample      2016-05-17 11:46:58.379795464 -0400
285 +++ /etc/nsd/nsd.conf      2016-05-18 07:06:14.861829191 -0400
286 @@ -23,6 +23,9 @@
287         # ip-address: 1.2.3.4
288         # ip-address: 1.2.3.4@5678
289         # ip-address: 12fe::8ef0
290 +
291         ip-address: 192.168.3.99
292 +
293         ip-address: ::1
294         ip-address: 127.0.0.
295         # Allow binding to non local addresses. Default no.
296         # ip-transparent: no
297
298 @@ -62,7 +65,7 @@

```

```

296
297     # the database to use
298     # if set to "" then no disk-database is used, less memory usage.
299 -     # database: "/var/db/nsd/nsd.db"
300 +     database: ""
301
302     # log messages to file. Default to stderr and syslog (with
303     # facility LOG_DAEMON). stderr disappears when daemon goes to bg.
304 @@ -141,7 +144,7 @@
305 remote-control:
306     # Enable remote control with nsd-control(8) here.
307     # set up the keys and certificates with nsd-control-setup.
308 -     # control-enable: no
309 +     control-enable: yes
310
311     # what interfaces are listened to for control, default is on localhost.
312     # control-interface: 127.0.0.1
313 @@ -249,4 +252,10 @@
314         # zonefile: "example.com.zone"
315         # request-xfr: 192.0.2.1 example.com.key
316
317 -
318 +pattern:
319 +     name: "local-signed"
320 +     zonefile: "/var/opendnssec/signed/%s"
321 +
322 +zone:
323 +     name: "nev1.dnslab.nccoe.nist.gov"
324 +     include-pattern: "local-signed"
325
326
327 [rdolmans@nsd ~]$ sudo groupadd -r nsd
328 [rdolmans@nsd ~]$ sudo useradd -r -g nsd -s /sbin/nologin -c "nsd daemon" nsd
329
330 # Make user nsd the owner of the nsd db and run directories
331 [rdolmans@nsd ~]# sudo chown nsd:nsd /var/db/nsd/
332 [rdolmans@nsd ~]# sudo chown nsd:nsd /var/run/nsd
333
334 # Start NSD
335 [rdolmans@nsd ~]$ sudo nsd-control start
336
337 # Export DS
338 [rdolmans@nsd ~]$ sudo ods-enforcer key export --zone nev1.dnslab.nccoe.nist.gov --ds
339 ;ready KSK DS record (SHA1):
340 nev1.dnslab.nccoe.nist.gov.      3600      IN      DS      35674 8 1
341 79ee1e53ce23658b6d5632297336b3067a80e329
342 ;ready KSK DS record (SHA256):
343 nev1.dnslab.nccoe.nist.gov.      3600      IN      DS      35674 8 2
344 0bd77d723e0a6d602a82bf0173a32a8286cfa4d602100e716192425544fb43a2
345 key export completed in 0 seconds.

```

```

346
347
348 Generate key + selfsigned cert:
349
350 [rdolmans@unbound cert]$ sudo openssl req -newkey rsa:2048 -nodes \
351 -keyout nev1.dnslab.nccoe.nist.gov.key -x509 -days 365 -out nev1.dnslab.nccoe.nist.gov.crt
352 Generating a 2048 bit RSA private key
353 .....+++
354 .....+++
355 writing new private key to 'nev1.dnslab.nccoe.nist.gov.key'
356 -----
357 You are about to be asked to enter information that will be incorporated into your certificate
358 request.
359 What you are about to enter is what is called a Distinguished Name or a DN.
360 There are quite a few fields but you can leave some blank
361 For some fields there will be a default value,
362 If you enter '.', the field will be left blank.
363 -----
364 Country Name (2 letter code) [XX]:NL
365 State or Province Name (full name) []:
366 Locality Name (eg, city) [Default City]:Amsterdam
367 Organization Name (eg, company) [Default Company Ltd]:NLnet Labs
368 Organizational Unit Name (eg, section) []:
369 Common Name (eg, your name or your server's hostname) []:nev1.dnslab.nccoe.nist.gov
370 Email Address []:
371
372
373 # Generate TLSA record for cert:
374
375 [rdolmans@unbound cert]$ ldns-dane create nev1.dnslab.nccoe.nist.gov 25 3 1 1 -c
376 nev1.dnslab.nccoe.nist.gov.crt
377 _25._tcp.nev1.dnslab.nccoe.nist.gov. 3600 IN TLSA 3:
378 1 1 0e8f0af01ea3c87bb5647de3f36cd7ab1eedf5ae466edf5a8800f6174884f60d
379
380 # Add TLSA and MX records to zone:
381
382 [rdolmans@nsd unsigned]$ diff -u nev1.dnslab.nccoe.nist.gov.old nev1.dnslab.nccoe.nist.gov
383 --- nev1.dnslab.nccoe.nist.gov.old      2016-05-31 10:13:17.728379254 -0400
384 +++ nev1.dnslab.nccoe.nist.gov 2016-05-31 10:13:21.403379256 -0400
385 @@ -9,7 +9,10 @@
386
387
388          NS      ns.nev1.dnslab.nccoe.nist.gov.
389          A       192.168.3.99
390 +         MX      10 192.168.3.98
391
392
393          TXT "dnslab test zone."
394
395

```

```
396 ns      IN      A      192.168.3.99
397 +
398 +_25._tcp  IN      TLSA      3 1 1 0e8f0af01ea3c87bb5647de3f36cd7ab1eedf5ae466edf5a8800f6174884f60d
399
400 # Resign
401 [rdolmans@nsd unsigned]$ sudo ods-signer sign nev1.dnslab.nccoe.nist.gov
402 Zone nev1.dnslab.nccoe.nist.gov scheduled for immediate re-sign.
403
```

Appendix G Microsoft Installation for the NCCoE

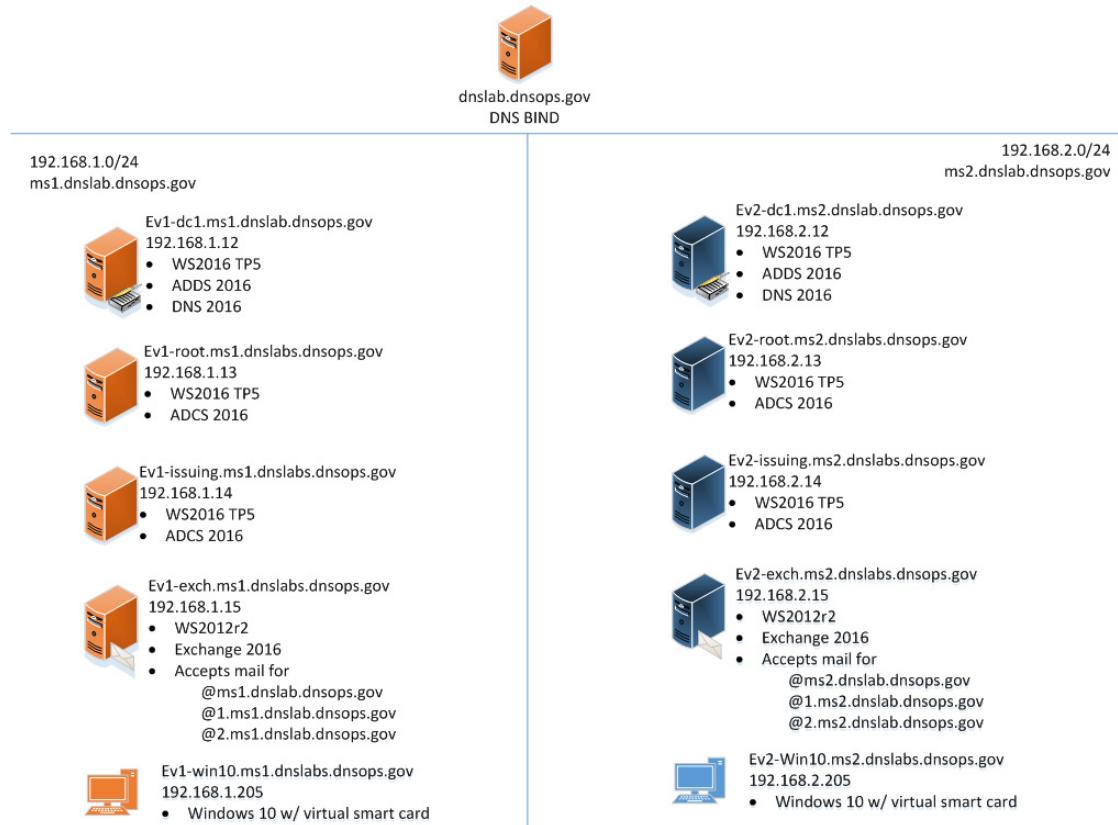
3 The following log captures the installation and configuration process for Microsoft system and
4 applications software for the NCCoE's DNS-Based Email Security project. Please note that the IP
5 addresses, domain names, and mail addresses are for the NCCoE laboratory and must not be
6 used in actual implementations.

7 G.1 Microsoft Server

8 Two Microsoft Active Directory domains were built for this project. MS1.DNSLAB.DNSOPS.GOV
9 and MS2.DNSLAB.DNSOPS.GOV domains. Two versions of Windows Server were used.
10 Windows Server 2016 Technical Preview 5, Standard GUI edition (WS2016TP5) which is
11 available from the Microsoft Evaluation Center (<https://www.microsoft.com/en-us/evalcenter/evaluate-windows-server-technical-preview>); and Active Directory Domain Services with
12 integrated Domain Name Services and Certificate Services run on WS2016TP5. Currently,
13 Exchange 2016 runs on Windows Server 2012R2 due to Exchange requirements ([https://technet.microsoft.com/en-us/library/aa996719\(v=exchg.160\).aspx](https://technet.microsoft.com/en-us/library/aa996719(v=exchg.160).aspx)).
14

15 The procession of Microsoft Services to be installed and configured is as follows:
16

- 17 1. Active Directory Domain Services
- 18 2. Active Directory Certificate Services - Root Certification Authority
- 19 3. Active Directory Certificate Services - Issuing Certification Authority
- 20 4. Active Directory Domain Name Services
- 21 5. Exchange 2016



22

G.2 Active Directory Domain Services

The following procedures were used for the creation of the MS1.DNSLAB.DNSOPS.GOV Active Directory domain on the EV1-DC1.MS1.DNSLAB.DNSOPS.GOV WS2016TP5 server.

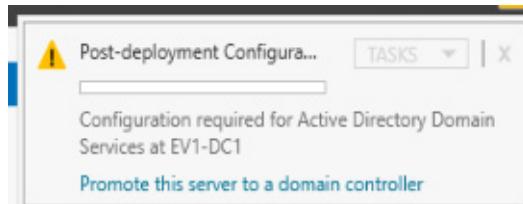
1. Statically assign IP address of the Domain Controller. This domain controller serves as the DNS server for the MS1.DNSLAB.DNSOPS.GOV Active directory domain:
 - a. IP Address: 192.168.1.12
 - b. Netmask: 255.255.255.0
 - c. Gateway: 192.168.1.1
 - d. DNS Server 192.168.1.12
2. Install Active Directory Domain Services (ADDS) role:
 - a. **Server Manager -> Manage -> Add Roles and Features**
 - b. **Installation type -> Role-based or feature based installation**
 - c. **Server Selection -> local server**
 - d. **Server Roles -> Select Active Directory Domain Services**, accept the Features to be added with the installation of ADDS.
 - e. On the **Features** selection menu click **Next**.

39 f. Click **Install**.

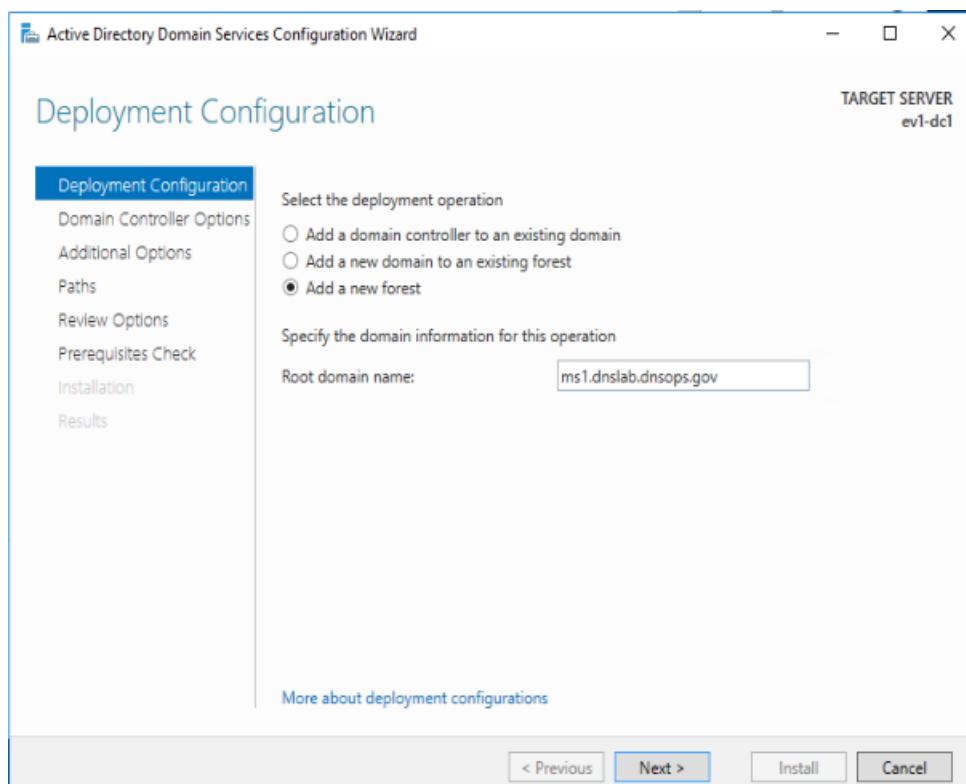
40 g. Once installation is complete click **Close**.

41 3. Configure the Active Directory Domain Services.

42 a. In Server Manager click the **exclamation mark** underneath the flag icon and click on **Promote this server to a domain controller**.



44 45 b. **Deployment Configuration -> Add a new forest** and specify the root name of **MS1.DNSLAB.DNSOPS.GOV**.



47 48 c. In **Domain Controller Options** select the defaults and set the **Directory Services Restore Mode (DSRM) password**.

49 d. DNS Options - parent zone could not be found, click **Next**.

50 e. The NetBios domain name will default to the lowest level of the FQDN of the Forest, i.e. **MS1**.

51 f. Accept the default paths for the ADDS Database, Log and SysVol folders. If running on a virtual machine, follow the recommended practice of the virtualization host.

55 g. In the Prerequisites Check you will be notified that the DNS cannot be delegated. The
 56 DNS server will be hosted on this domain controller.

57 **G.3 Active Directory Certificate Services: Microsoft
 58 Certificate Authority**

59 Windows Server 2016 TP5 Active Directory Certificate Services (ADCS) serves as the Public Key
 60 Infrastructure for the MS1.DNSLAB.DNSOPS.GOV namespace. It is a two-tier hierarchy with
 61 EV1-ROOT.MS1.DNSLAB.DNSOPS.GOV as the root Certification Authority (CA) trust point, and
 62 EV1-ISSUING.MS1.DNSLAB.DNSOPS.GOV as the domain joined enterprise issuing CA.

63 **G.3.1 Root CA Installation**

64 The installation of Active Directory Certificate Services must be performed by an enterprise
 65 administrator.

66 1. **Copy CAPolicy.inf to the c:\windows directory:**

```
67 ; NCCoE DANE DNSSEC Building Block
68
69 [Version]
70 Signature= "$Windows NT$"
71
72 ; Configures CA to allow only a single tier of CAs below it
73 [BasicConstraintsExtension]
74 PathLength = 1
75
76 ; Allows all issuance policies, sets HTTP pointer for CPS
77 [PolicyStatementExtension]
78 Policies = AllIssuancePolicy, LegalPolicy
79 Critical = 0
80
81 [AllIssuancePolicy]
82 OID = 2.5.29.32.0
83
84 [LegalPolicy]
85 OID = 1.1.1.1.1
86 Notice = "http://pki.ms1.dnslab.dnsops.gov/CPS.htm"
87 URL = "http://pki.ms1.dnslab.dnsops.gov/CPS.htm"
88
89 ; Sets key renewal and CRL publication parameters
90 [Certsrv_Server]
91 RenewalKeyLength = 4096
92 RenewalValidityPeriod = Years
93 RenewalValidityPeriodUnits = 20
94 CRLPeriod = days
95 CRLPeriodUnits = 180
96 CRLDeltaPeriodUnits = 0
```

```

97 CRLDeltaPeriod = days
98
99 ; Makes the CDP and AIA pointer for the root CA cert blank
100 [CRLDistributionPoint]
101 Empty = True
102
103 [AuthorityInformationAccess]
104 Empty = True
105
106 ; NCCoE DANE DNSSEC Building Block
107
108 [Version]
109 Signature= "$Windows NT$"
110
111 ; Configures CA to allow only a single tier of CAs below it
112 [BasicConstraintsExtension]
113 PathLength = 1
114
115 ; Allows all issuance policies, sets HTTP pointer for CPS
116 [PolicyStatementExtension]
117 Policies = AllIssuancePolicy, LegalPolicy
118 Critical = 0
119
120 [AllIssuancePolicy]
121 OID = 2.5.29.32.0
122
123 [LegalPolicy]
124 OID = 1.1.1.1.1
125 Notice = "http://pki.ms1.dnslab.dnsops.gov/CPS.htm"
126 URL = "http://pki.ms1.dnslab.dnsops.gov/CPS.htm"
127
128 ; Sets key renewal and CRL publication parameters
129 [Certsrv_Server]
130 RenewalKeyLength = 4096
131 RenewalValidityPeriod = Years
132 RenewalValidityPeriodUnits = 20
133 CRLPeriod = days
134 CRLPeriodUnits = 7
135 CRLDeltaPeriodUnits = 0
136 CRLDeltaPeriod = days
137
138 ; Makes the CDP and AIA pointer for the root CA cert blank
139 [CRLDistributionPoint]
140 Empty = True
141
142 [AuthorityInformationAccess]
143 Empty = True
144

```

145 2. **Server Manager -> Manage -> Add Roles and Features.**

146 3. **Installation type -> Role-based or feature based** installation.

147 4. **Server Selection -> local server.**

148 5. **Server Roles -> Select Active Directory Certificate Services**, accept the Features to be added with the installation of ADCS.

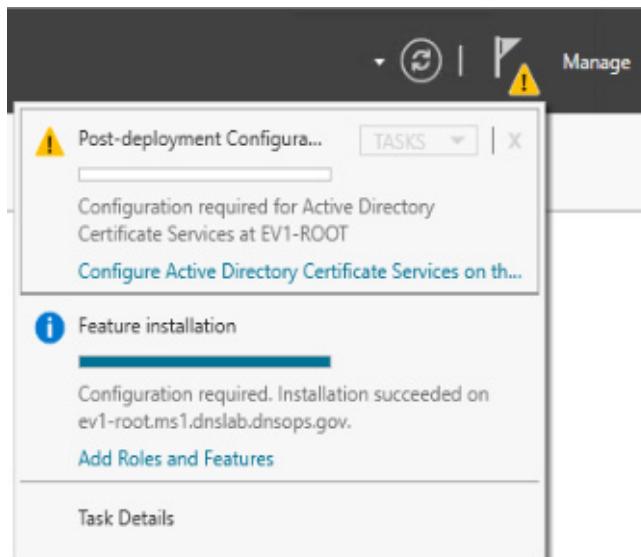
149 6. On the **Features** selection menu click **Next**.

150 7. Click **Install**.

151 8. Once installation is complete click **Close**.

153 **G.3.1.1 Configure Root CA**

154 1. **Run post install configuration wizard**, click on **Configure Active Directory Certificate Services** link:



156 2. **Select Role Services to configure -> select Certification Authority.**

157 3. **Setup Type = Standalone CA.**

158 4. **CA Type = Root CA.**

159 5. **Private Key = Create a new private key.**

160 6. **Cryptography:**

161 a. **Cryptographic provider -> RSA#Microsoft Software Key Storage Provider**

162 b. **Hashing Algorithm = SHA256**

163 c. **Key Length 2048**

164 7. **CA Name = EV1-Root**

165 8. Once completed, **run the post install script**.

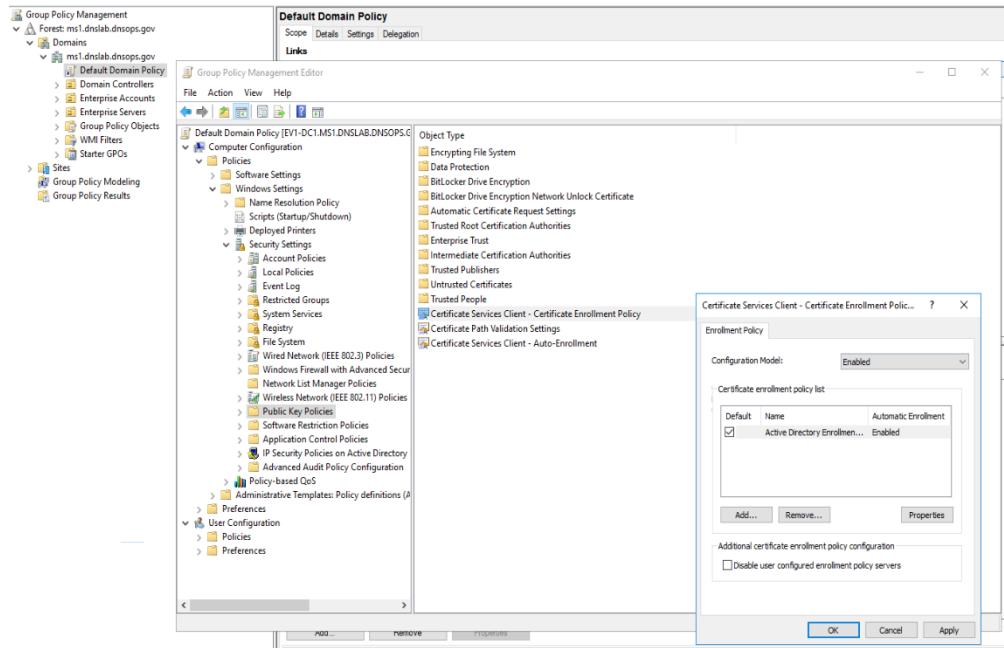
```

167 :: NCCoE DANE DNSSEC Building Block
168
169 :: Declares configuration NC
170 certutil -setreg CA\DSConfigDN CN=Configuration,DC=ms1,DC=dnslab,DC=dnsops,DC=gov
171
172 :: Defines CRL publication intervals
173 certutil -setreg CA\CRLPeriodUnits 7
174 certutil -setreg CA\CRLPeriod "Days"
175 certutil -setreg CA\CRLDeltaPeriodUnits 0
176 certutil -setreg CA\CRLDeltaPeriod "Days"
177
178 :: Specifies CDP attributes
179 certutil -setreg CA\CRLPublicationURLs
180 "65:%windir%\system32\CertSrv\CertEnroll\%3%8%9.crl\n6:http://pki.ms1.dnslab.dnsops.gov/
181 %%3%8%9.crl\n14:ldap://CN=%7%,CN=%2,CN=CDP,CN=Public Key Services,CN=Services,%%6%10\n"
182
183 :: Specifies AIA attributes
184 certutil -setreg CA\CACertPublicationURLs
185 "1:%windir%\system32\CertSrv\CertEnroll\%7.crt\n2:http://pki.ms1.dnslab.dnsops.gov/
186 %%7.crt\n3:ldap://CN=%7,CN=AIA,CN=Public Key Services,CN=Services,%%6%11\n"
187
188 :: Enables auditing all events for the CA
189 certutil -setreg CA\AuditFilter 127
190
191 :: Sets validity period for issued certificates
192 certutil -setreg CA\ValidityPeriodUnits 10
193 certutil -setreg CA\ValidityPeriod "Years"
194
195 :: Restarts Certificate Services
196 net stop certsrv & net start certsrv
197
198 :: Republishes the CRL; sometimes this gets an access denied (error 5) because the service is not
199 ready after restart, in this case, manually execute
200 certutil -crl

```

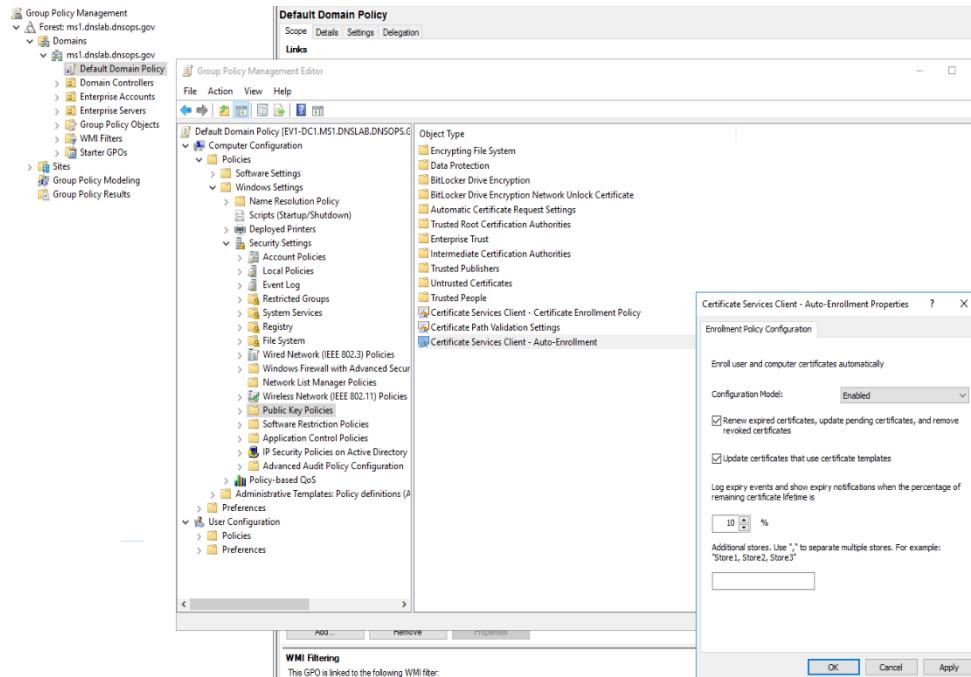
201 G.3.1.2 Enable Certificate Services Auto Enrollment within the Active Directory Domain

- 202 1. Log on to the domain controller EV1-DC1.MS1.DNSLAB.DNSOPS.GOV.
- 203 2. Start **Group Policy Management console** (gpmc.msc).
- 204 3. Navigate to the **Default Domain Policy**.
- 205 4. Within the Default Domain Policy go to **Computer Configuration -> Policies -> Windows**
Settings -> Security Settings -> Public Key Policies
- 207 5. Select the **Certificate Services Client - Certificate Enrollment Policy** setting.
- 208 6. Set to **Enabled**, ensure the **default Active Directory Enrollment Policy** is selected and click
OK.



210

7. Select Certificate Services Client - Auto-Enrollment setting.



212

8. Set Configuration Model to Enabled.

9. Enable Renew Expired Certificates and Update certificates that use certificate templates radio buttons.

216 G.3.2 Issuing a CA Installation

- 217 1. Start administrative command prompt as an Enterprise Administrator.
- 218 2. Publish the EV1-Root CA certificate to Active Directory for dissemination to all systems
- 219 within the MS1.DNSLAB.DNSOPS.GOV Active Directory domain. From an administrative
- 220 command prompt, type `certutil -dspublish -f ev1-root.crt rootca`.
- 221 3. From the administrative command prompt, type `certutil -pulse` followed by `gpupdate /`
- 222 `force`.
- 223 4. Copy **CAPolicy.inf** to the **c:\windows** directory.

```

224
225 ; NCCoE DANE DNSSEC Building Block
226
227 [Version]
228 Signature= "$Windows NT$"
229
230 ; Allows all issuance policies, sets HTTP pointer for CPS
231 [PolicyStatementExtension]
232 Policies = AllIssuancePolicy, LegalPolicy
233 Critical = 0
234
235 [AllIssuancePolicy]
236 OID = 2.5.29.32.0
237
238 [LegalPolicy]
239 OID = 1.1.1.1.1
240 Notice = "http://pki.ms1.dnslab.dnsops.gov/cps.htm"
241 URL = "http://pki.ms1.dnslab.dnsops.gov/CPS.htm"
242
243 ; Sets key renewal and CRL publication parameters
244 [certsrv_server]
245 renewkeylength = 2048
246 RenewalValidityPeriodUnits = 10
247 RenewalValidityPeriod = years
248 CRLPeriod = hours
249 CRLPeriodUnits = 36
250 CRLDeltaPeriod = hours
251 CRLDeltaPeriodUnits = 0
252

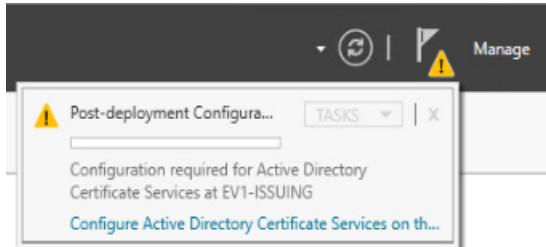
```

- 253 5. **Server Manager -> Manage -> Add Roles and Features.**
- 254 6. **Installation type -> Role-based or feature based installation.**
- 255 7. **Server Selection -> local server.**
- 256 8. **Server Roles -> Select Active Directory Certificate Services**, accept the Features to be
- 257 added with the installation of ADCS.
- 258 9. **Features = Certification Authority and Certification Authority Web Enrollment** (this will
- 259 add the required IIS features).
- 260 10. On the **Features** selection menu click **Next**.

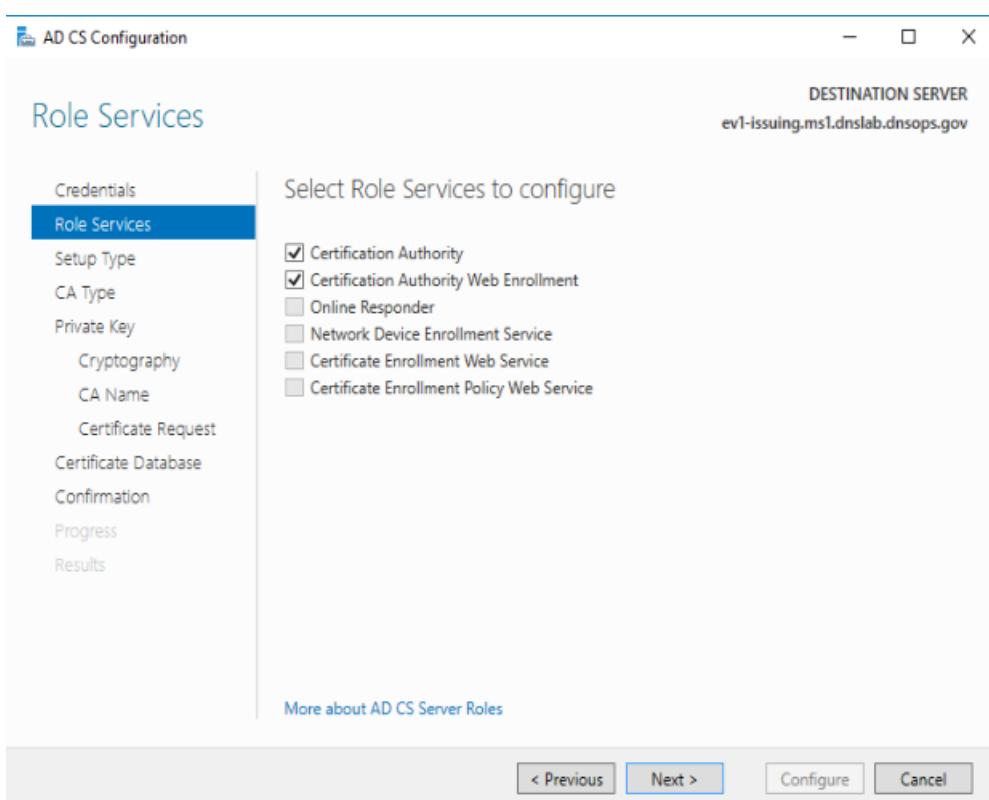
261 11. Click **Install**.

262 12. Once installation is complete click **Close**.

263 13. Run the **Post-Deployment configuration for the ADCS role**.



264 14. Select both **Certification Authority** and **Certification Authority Web Enrollment**.



265 15. Setup Type = **Enterprise CA**

266 16. CA Type = **Subordinate CA**

267 17. **Create new key** (same as above).

268 18. CA Name = **EV1-Issuing**

269 a. Private Key = **Create a new private key**

270 b. Cryptography:

271 i. Cryptographic provider -> **RSA#Microsoft Software Key Storage Provider**

272 ii. Hashing Algorithm = **SHA256**

275 iii. Key Length **2048**

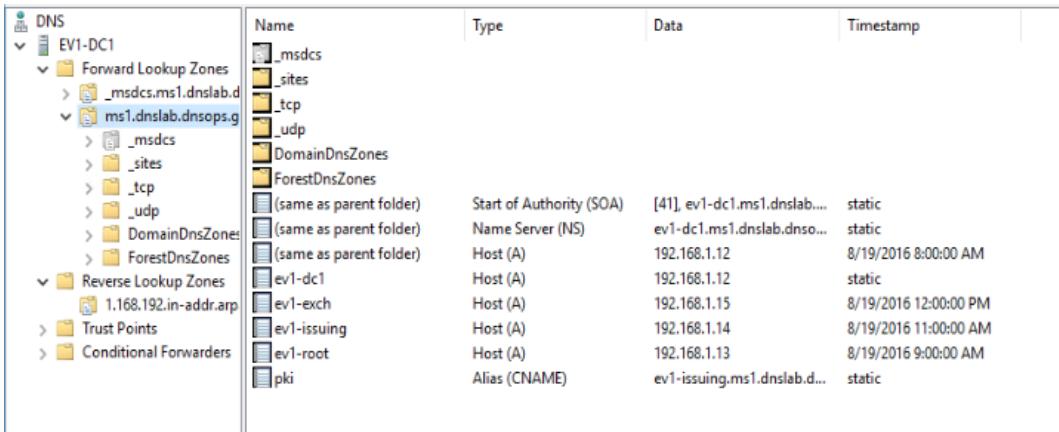
276 19. **Save** the request file to the **c:** drive.

277 20. **Copy** request file to **root ca**.

278 21. On Root CA, **issue certificate**.

279 22. Import **ev1-issuing.ca** into the **Certification Authority**.

280 23. Create a **CNAME** record for **PKI.MS1.DNSLAB.DNSOPS.GOV** to point to **ev1-issuing.ms1.dnslab.dnsops.gov**.



The screenshot shows the Windows DNS Manager interface. On the left, a tree view of the DNS structure under 'EV1-DC1' shows 'Forward Lookup Zones' and 'Reverse Lookup Zones'. The 'Forward Lookup Zones' section contains '_msdcsls', '_sites', '_tcp', '_udp', 'DomainDnsZones', and 'ForestDnsZones'. The 'Reverse Lookup Zones' section contains '1.168.192.in-addr.arp'. On the right, a table lists DNS records with columns for Name, Type, Data, and Timestamp.

Name	Type	Data	Timestamp
_msdcsls	Start of Authority (SOA)	[41], ev1-dc1.ms1.dnslab....	static
_sites	Name Server (NS)	ev1-dc1.ms1.dnslab.dns... ev1-issuing.ms1.dnslab.dns... ev1-root.ms1.dnslab.dns... pki.ms1.dnslab.dns...	static
_tcp	Host (A)	192.168.1.12	8/19/2016 8:00:00 AM
_udp	Host (A)	192.168.1.12	static
DomainDnsZones	Host (A)	192.168.1.15	8/19/2016 12:00:00 PM
ForestDnsZones	Host (A)	192.168.1.14	8/19/2016 11:00:00 AM
(same as parent folder)	Host (A)	192.168.1.13	8/19/2016 9:00:00 AM
ev1-dc1	Alias (CNAME)	ev1-issuing.ms1.dnslab.d... ev1-root.ms1.dnslab.dns... pki.ms1.dnslab.dns...	static
ev1-exch	Host (A)	192.168.1.15	8/19/2016 12:00:00 PM
ev1-issuing	Host (A)	192.168.1.14	8/19/2016 11:00:00 AM
ev1-root	Host (A)	192.168.1.13	8/19/2016 9:00:00 AM
pki	Alias (CNAME)	ev1-issuing.ms1.dnslab.d... ev1-root.ms1.dnslab.dns... pki.ms1.dnslab.dns...	static

282

283 24. Open **Internet Information Service Manager**.

284 25. Go to the **Default Web Site**.

285 26. Bindings: edit the existing default HTTP binding and add **pki.ms1.dnslab.dnsops.gov**.

286 27. Click on the **Filter requests** -> Select **Allow File name Extension** and add **.crl**, **.crt** and **.cer**.

287 28. From an administrative command prompt type **iisreset**.

288 29. On the Issuing CA run the post install script.

```

289 :: NCCoE DANE DNSSEC Building Block
290
291 :: Declares configuration NC
292 certutil -setreg CA\DSConfigDN CN=Configuration,DC=MS1,DC=DNSLAB,DC=DNSOPS,DC=GOV
293
294 :: Defines CRL publication intervals
295 certutil -setreg CA\CRLPeriodUnits 3
296 certutil -setreg CA\CRLPeriod "days"
297 certutil -setreg CA\CRLDeltaPeriodUnits 0
298 certutil -setreg CA\CRLDeltaPeriod "Hours"
299
300 :: Specifies CDP attributes
301 certutil -setreg CA\CRLPublicationURLS
302 "65:%windir%\system32\CertSrv\CertEnroll\%3%8%9.crl\n6:http://pki.ms1.dnslab.dnsops.gov/
303 %%3%%8%%9.crl\n79:ldap:///CN=%7%%8,CN=%2,CN=CDP,CN=Public Key Services,CN=Services,%%6%%10\n"
304
305 :: Specifies AIA attributes

```

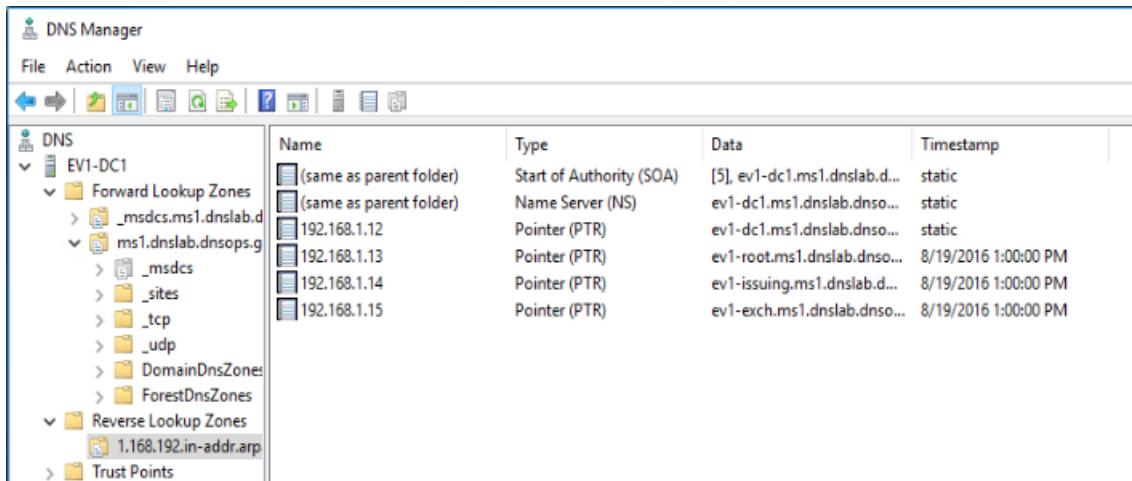
```

306 certutil -setreg CA\CACertPublicationURLs
307 "1:%windir%\system32\CertSrv\CertEnroll\%7.crt\n2:http://pki.ms1.dnslab.dnsops.gov/
308 %%7.crt\n3:ldap:///CN=%7,CN=AIA,CN=Public Key Services,CN=Services,%%6%%11\n"
309
310 :: Enables auditing all events for the CA
311 certutil -setreg CA\AuditFilter 127
312
313 :: Sets maximum validity period for issued certificates
314 certutil -setreg CA\ValidityPeriodUnits 5
315 certutil -setreg CA\ValidityPeriod "Years"
316
317 :: Restarts Certificate Services
318 net stop certsrv & net start certsrv
319
320 :: Republishes the CRL; sometimes this gets an access denied (error 5) because the service is not
321 ready after restart, in this case, manually execute
322 certutil -CRL

```

323 G.4 Microsoft Domain Name Services: DNS Domain 324 Server

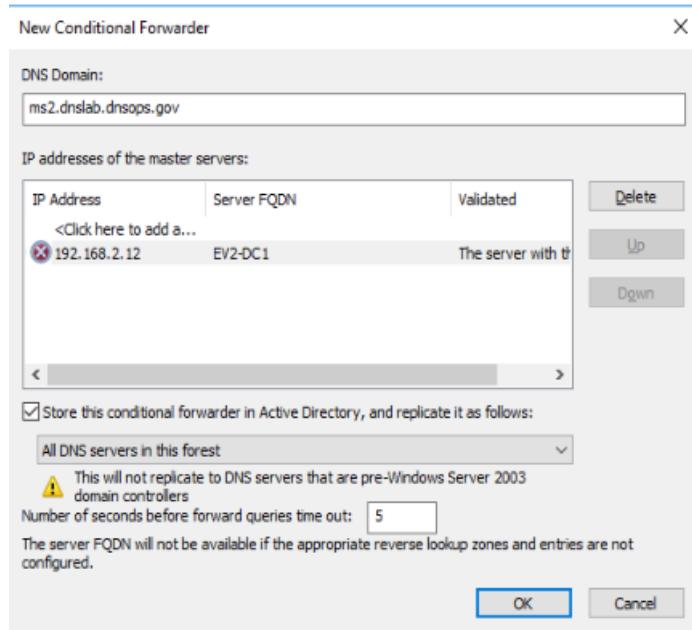
325 Active Directory Domain Services installation installs and configures the ms1.dnslab.dnsops.gov
326 Forward lookup zone. It is recommended to create a Reverse lookup zone for the subnets used.



The screenshot shows the Windows DNS Manager. The left pane displays the DNS tree structure under the 'EV1-DC1' domain controller. The 'Forward Lookup Zones' section contains the '_msdcs.ms1.dnslab.dnsops.g' zone, which is expanded to show subzones like '_msdcs', '_sites', '_tcp', '_udp', 'DomainDnsZones', and 'ForestDnsZones'. The 'Reverse Lookup Zones' section contains the '1.168.192.in-addr.ap' zone, which is also expanded to show subzones like 'Trust Points'. The right pane is a table showing DNS records for the 'ms1.dnslab.dnsops.g' zone. The table has columns for 'Name', 'Type', 'Data', and 'Timestamp'. The data includes:

Name	Type	Data	Timestamp
(same as parent folder)	Start of Authority (SOA)	[5], ev1-dc1.ms1.dnslab.d...	static
(same as parent folder)	Name Server (NS)	ev1-dc1.ms1.dnslab.dnsop...	static
192.168.1.12	Pointer (PTR)	ev1-dc1.ms1.dnslab.dnsop...	static
192.168.1.13	Pointer (PTR)	ev1-root.ms1.dnslab.dnsop...	8/19/2016 1:00:00 PM
192.168.1.14	Pointer (PTR)	ev1-issuing.ms1.dnslab.d...	8/19/2016 1:00:00 PM
192.168.1.15	Pointer (PTR)	ev1-exch.ms1.dnslab.dnsop...	8/19/2016 1:00:00 PM

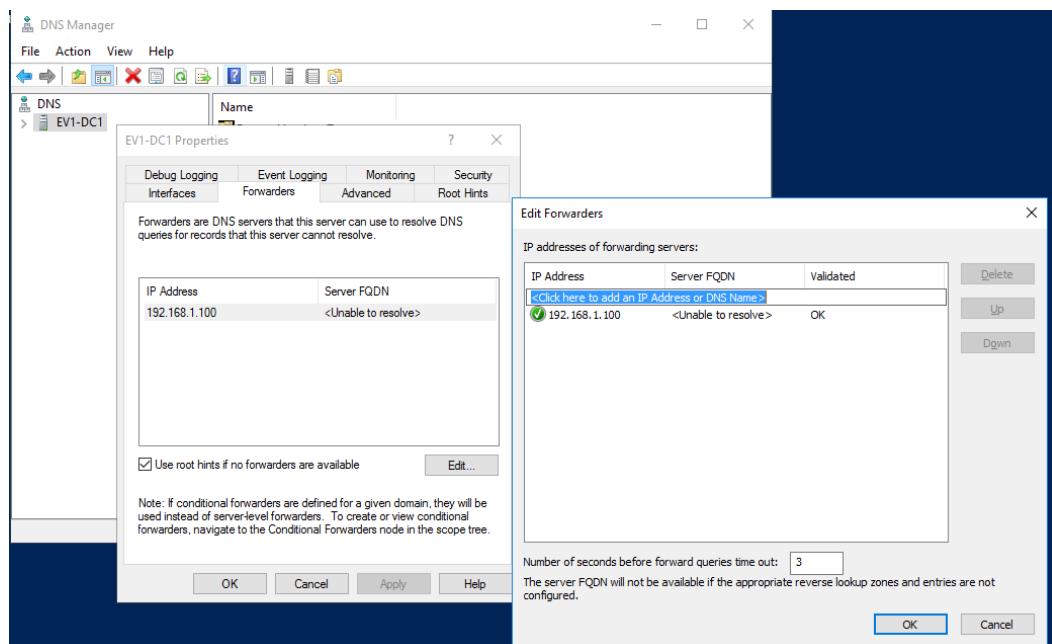
327 1. Create a conditional forwarder for the other name spaces:



329

330

2. Create forwarded to dnslab.dnsops.gov.



331

G.5 Microsoft Exchange

332 Exchange 2016 was installed on a Windows Server 2012R2 Standard (Server with a GUI).
 333 Exchange 2016 is currently not supported on Windows Server 2016 Technical Preview 2016
[https://technet.microsoft.com/en-us/library/aa996719\(v=exchg.160\).aspx](https://technet.microsoft.com/en-us/library/aa996719(v=exchg.160).aspx).

334 Exchange 2016 prerequisites can be found here: [https://technet.microsoft.com/en-us/library/bb691354\(v=exchg.160\).aspx](https://technet.microsoft.com/en-us/library/bb691354(v=exchg.160).aspx).

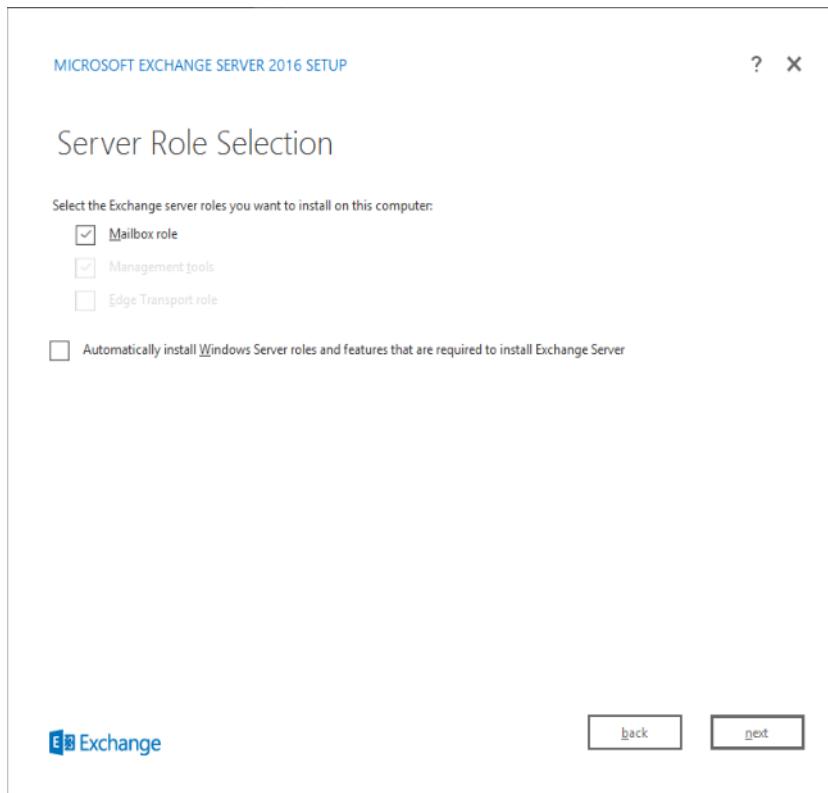
338 Download for .Net 4.5.2: <https://www.microsoft.com/en-us/download/details.aspx?id=42642>.

339 1. Install the Remote Tools Administration Pack using the following powershell command:
 340 `Install-WindowsFeature RSAT-ADDS.`

341 2. Install Exchange 2016 prerequisites with the following powershell command:
 342 `Install-WindowsFeature AS-HTTP-Activation, Desktop-Experience, NET-`
 343 `Framework-45-Features, RPC-over-HTTP-proxy, RSAT-Clustering, RSAT-`
 344 `Clustering-CmdInterface, RSAT-Clustering-Mgmt, RSAT-Clustering-PowerShell,`
 345 `Web-Mgmt-Console, WAS-Process-Model, Web-Asp-Net45, Web-Basic-Auth, Web-`
 346 `Client-Auth, Web-Digest-Auth, Web-Dir-Browsing, Web-Dyn-Compression, Web-`
 347 `Http-Errors, Web-Http-Logging, Web-Http-Redirect, Web-Http-Tracing, Web-`
 348 `ISAPI-Ext, Web-ISAPI-Filter, Web-Lgcy-Mgmt-Console, Web-Metabase, Web-Mgmt-`
 349 `Console, Web-Mgmt-Service, Web-Net-Ext45, Web-Request-Monitor, Web-Server,`
 350 `Web-Stat-Compression, Web-Static-Content, Web-Windows-Auth, Web-WMI,`
 351 `Windows-Identity-Foundation`

352 3. Perform Active Directory Schema update following the Technet article, "Prepare Active
 353 Directory and Domains": [https://technet.microsoft.com/en-us/library/bb125224\(v=exchg.160\).aspx](https://technet.microsoft.com/en-us/library/bb125224(v=exchg.160).aspx).

354 4. Install the **Mailbox role**.



356 5. Once the installation is completed go to the **Exchange Admin console**: <https://ev1-exch.ms1.dnslab.dnsops.gov/ECP>.

358 6. Create an **Internet send connector** following this Technet article: [https://technet.microsoft.com/en-us/library/jj657457\(v=exchg.160\).aspx](https://technet.microsoft.com/en-us/library/jj657457(v=exchg.160).aspx).

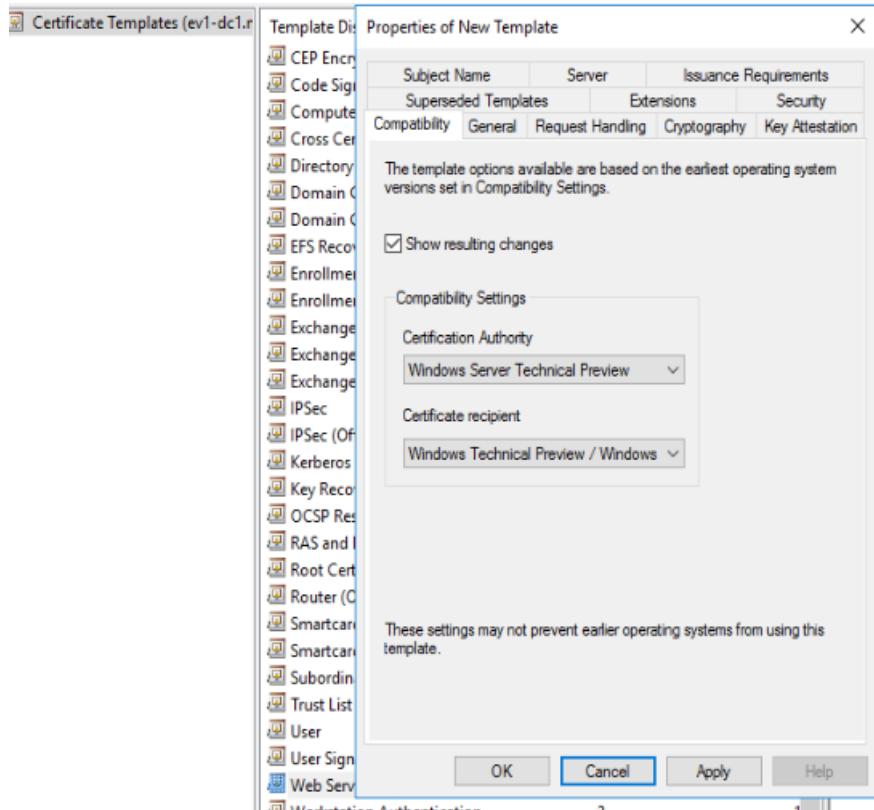
360 7. Create an **SSL certificate** for the Exchange services.

362 8. On the Issuing CA (ev1-issuing), open **Certification Authority -> Certificate Templates**.

363 9. **Right click -> Manage**.

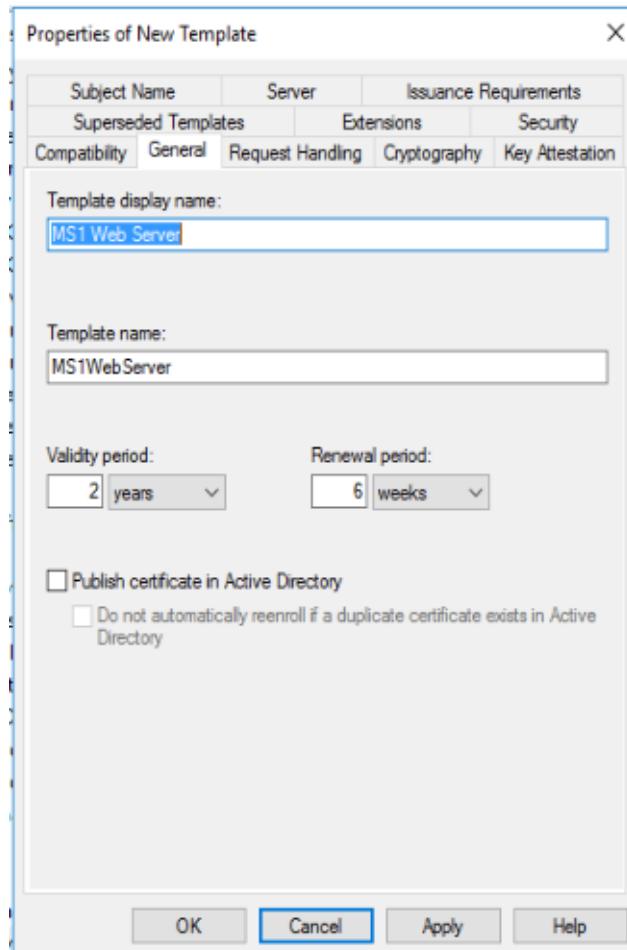
364 10. Right click on the **Web Server template** and select **duplicate**.

365 11. Compatibility = **Windows Server Technical Preview**



366

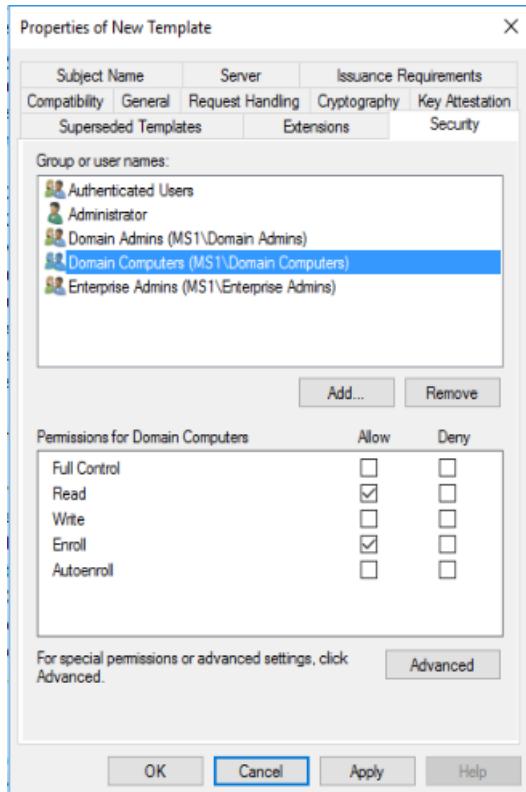
367 12. **General -> Template Display Name MS1 Web Server**



368

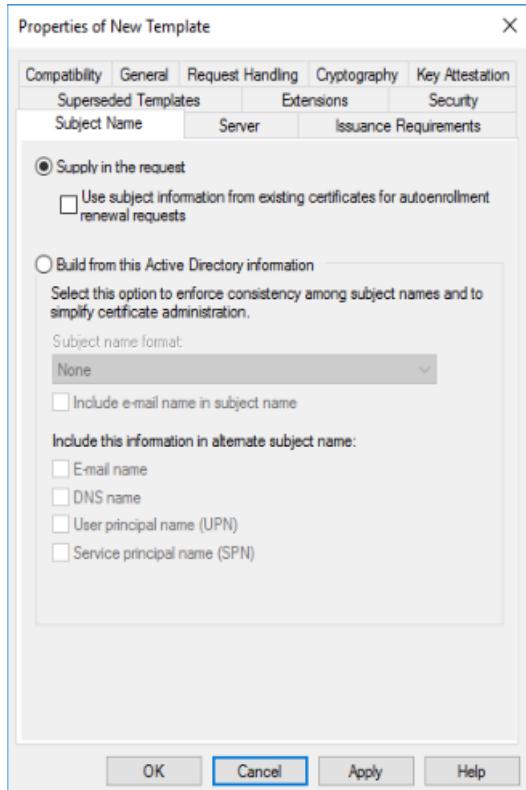
13. Security -> Domain Computers allowed to Enroll for certificate

369



370

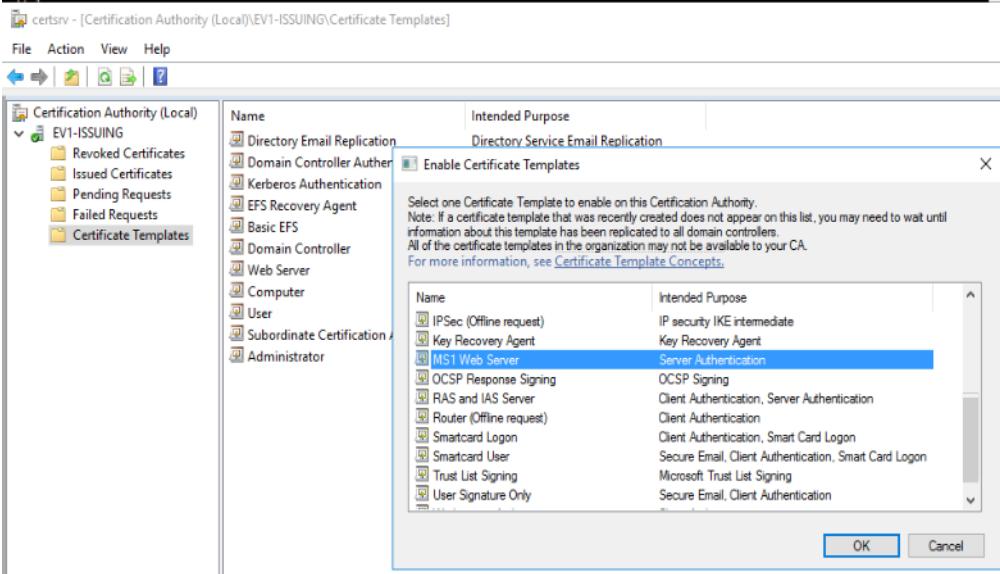
14. Subject Name -> Supply in Request



372

373 15. Click **OK** to save the new MS1 Web Server certificate template.

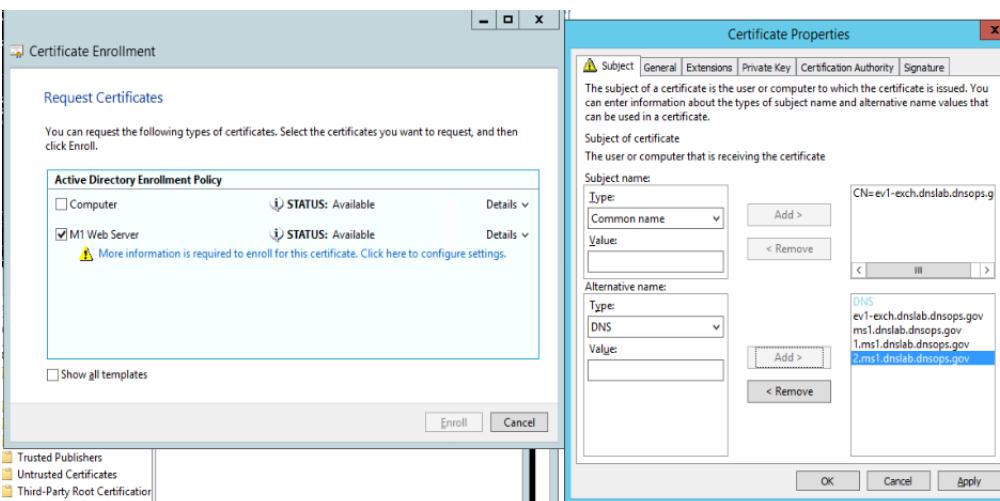
374 16. Back in the Certification Authority snap-in, right click on **Certificate Templates -> Certificate to Issue**, then select the **MS1 Web Server** certificate template.



376

377 17. On the Exchange server (ev1-exch), log on as an administrator and type **certlm.msc**.

378 18. Go to **Personal -> Certificates -> right click -> request new certificate**.



379

380 19. Subject Name: **Common Name = ev1-exch.ms1.dnslab.dnsops.gov**

381 20. Alternative Name: **DNS = ev1-exch.ms1.dnslab.dnsops.gov, ms1.dnslab.dnsops.gov, 1.ms1.dnslab.dnsops.gov, 2.ms1.dnslab.dnsops.gov**

382

383 21. Click **OK** and then select **enroll**.

384 22. Use this certificate to protect the Exchange services.

385 23. Within the Exchange Admin console (<https://ev1-exch.ms1.dnslab.dnsops.gov/ECP>), select **Server -> Certificates**, then change all services to use the issued SSL certificate.

servers databases database availability groups virtual directories certificates

Select server: ev1-exch.ms1.dnslab.dnsops.gov

+

NAME	STATUS	EXPIRES ON
Exchange SSL	Valid	8/19/2018
Microsoft Exchange Server Auth Certificate	Valid	8/5/2021
Microsoft Exchange	Valid	8/31/2021
WMSVC	Valid	8/17/2026

Exchange SSL

Certification authority-signed certificate
Issuer: CN=EV1-ISSUING, DC=ms1, DC=dnslab, DC=dnsops, DC=gov

Status
Valid
Expires on: 8/19/2018
Renew

Assigned to services
NONE

387

Exchange Certificate - Internet Explorer
https://ev1-exch.ms1.dnslab.dnsops.gov/ecp/CertMgmt/EditCertificate.aspx?pwmcid=3&ReturnObjectType=1&id=ev1-exch.ms1.dns1a

Exchange SSL

general services

Name: Exchange SSL

Status: Valid

Issuer: CN=EV1-ISSUING, DC=ms1, DC=dnslab, DC=dnsops, DC=gov

Expires on: 8/19/2018

Subject: CN=ev1-exch.dnslab.dnsops.gov

Subject Alternative Names:
ev1-exch.dnslab.dnsops.gov
ms1.dnslab.dnsops.gov
1.ms1.dnslab.dnsops.gov
2.ms1.dnslab.dnsops.gov

Thumbprint: CDE061915589A82EA0B1FCD915469188D4D030CB

Serial number: 32000000088AD81A5A47E1AC0C000100000008

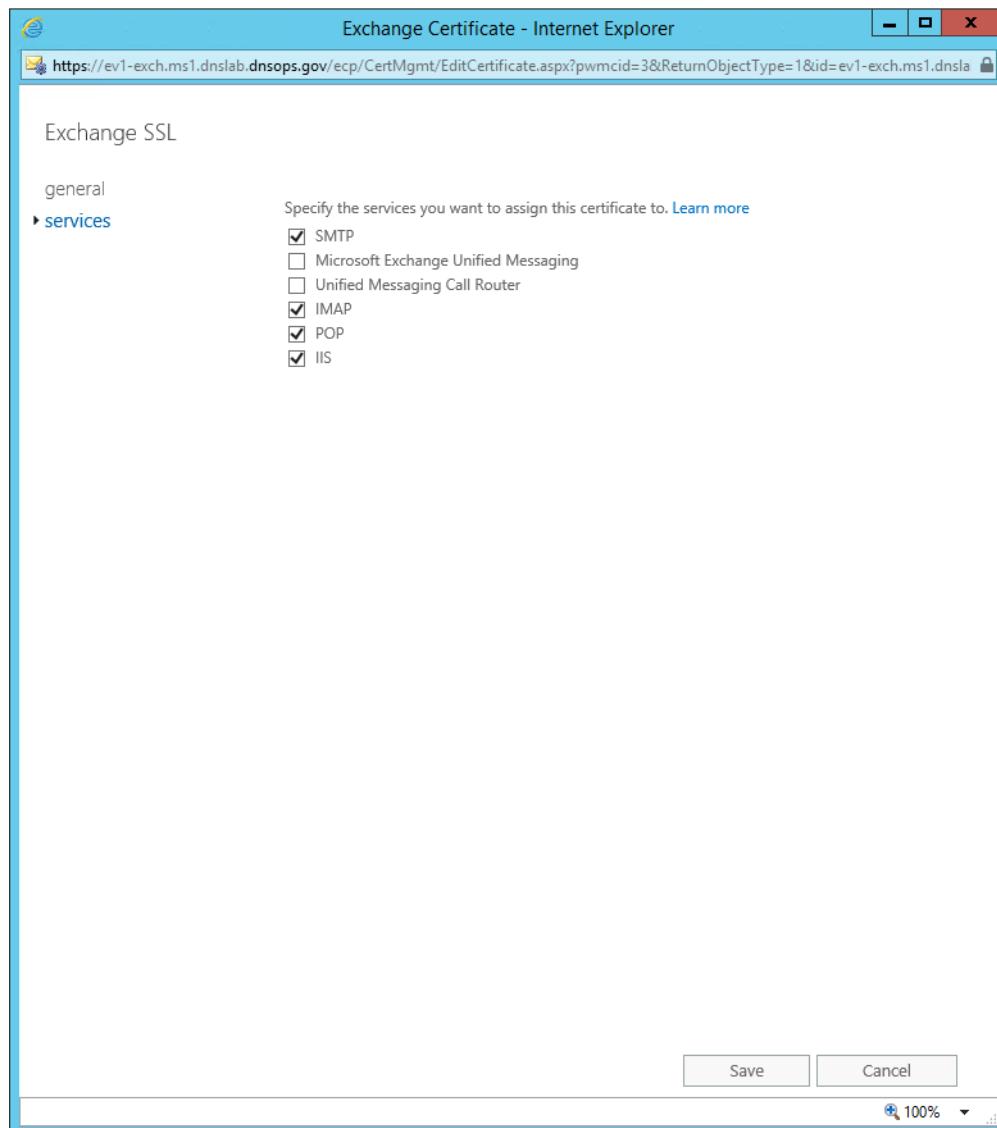
Public key size: 2048

Has private key: Yes

Save Cancel 100%

388

24. Select all the services **except** for Unified Messaging.



390

391 G.5.1 Generate the TLS DNS Record

392 1. Sign the ms1.dnslab.dnsops.gov zone by following the Technet article for enabling DNSSEC
 393 <https://technet.microsoft.com/en-us/library/hh831411.aspx>.

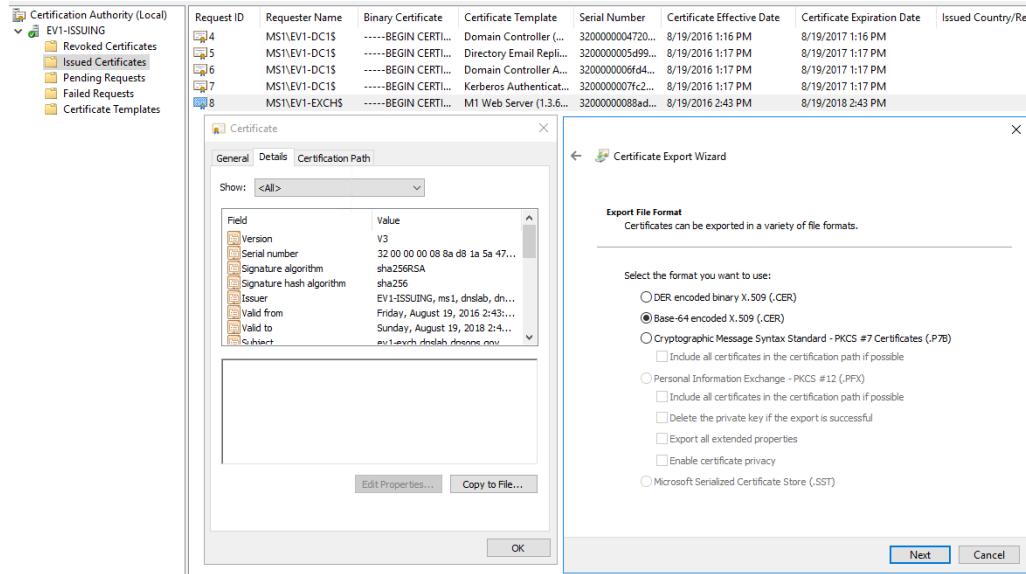
394 2. **Export the Exchange SSL certificate to a .cer file.** Find the certificate on the Issuing CA (ev1-
 395 issuing) within the **Issued Certificates** group.

Request ID	Requester Name	Binary Certificate	Certificate Template
4	MS1\EV1-DC1\$	-----BEGIN CERTI...	Domain Controller (...
5	MS1\EV1-DC1\$	-----BEGIN CERTI...	Directory Email Repli...
6	MS1\EV1-DC1\$	-----BEGIN CERTI...	Domain Controller A...
7	MS1\EV1-DC1\$	-----BEGIN CERTI...	Kerberos Authenticat...
8	MS1\EV1-EXCH\$	-----BEGIN CERTI...	M1 Web Server (1.3.6...

396

397

3. Click on the **Details** tab and select **Copy to File**. Save as a **base64 (.cer)** file.

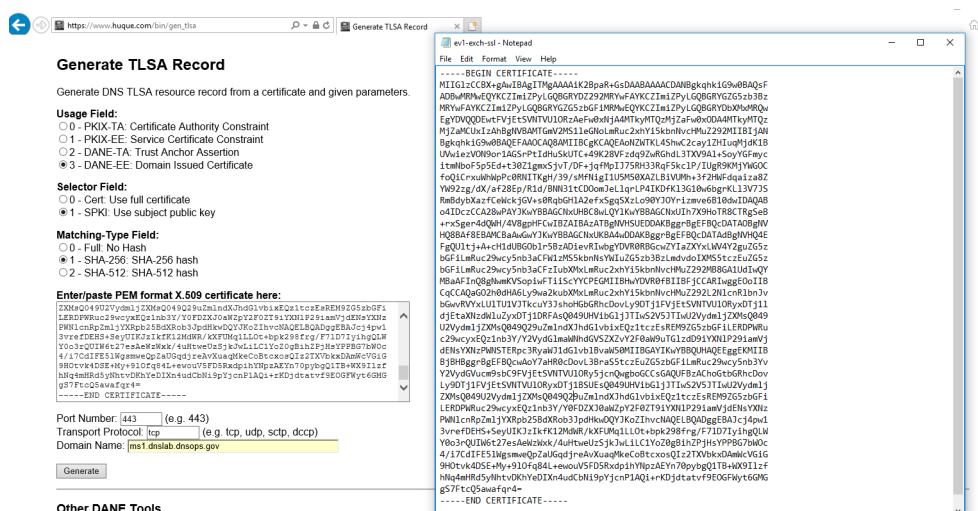


398

399 4. Go to https://www.huque.com/bin/gen_tlsa. Open the exported certificate into notepad, the **copy and paste** into the **Enter/paste PEM format X.509 certificate here** field.

400

401 5. Fill in the name space specific information.



402

403 6. Select **Generate** and the TLSA record string is presented back.

404 `_443._tcp.ms1.dnslab.dnsops.gov. IN TLSA 3 1 1`

405 `25d645a7bd304ae552c629ca5e7061a70f921afc4dd49c1ea0c8f22de6595be7`

406

407 7. To register this TLSA record within Windows Server 2016 Active Directory Domain Name
 408 Services, issue the following powershell command on the Domain Controller as
 409 Administrator:

```
410 add-dnsserverresourcerecord -TLSA -CertificateAssociationData
411 "25d645a7bd304ae552c629ca5e7061a70f921afc4dd49c1ea0c8f22de6595be7" -
412 CertificateUsage DomainIssuedCertificate -MatchingType Sha256Hash -Selector
413 FullCertificate -ZoneName ms1.dnslab.dnsops.gov -Name _25._tcp.ev1-
414 exch.ms1.dnslab.dnsops.gov.
```

415 8. To get the zone output, issue the following powershell command:

```
416 Resolve-DnsName ev1-dc1.ms1.dnslab.dnsops.gov -type soa -server ev1-dc1 -
417 DnssecOk
```

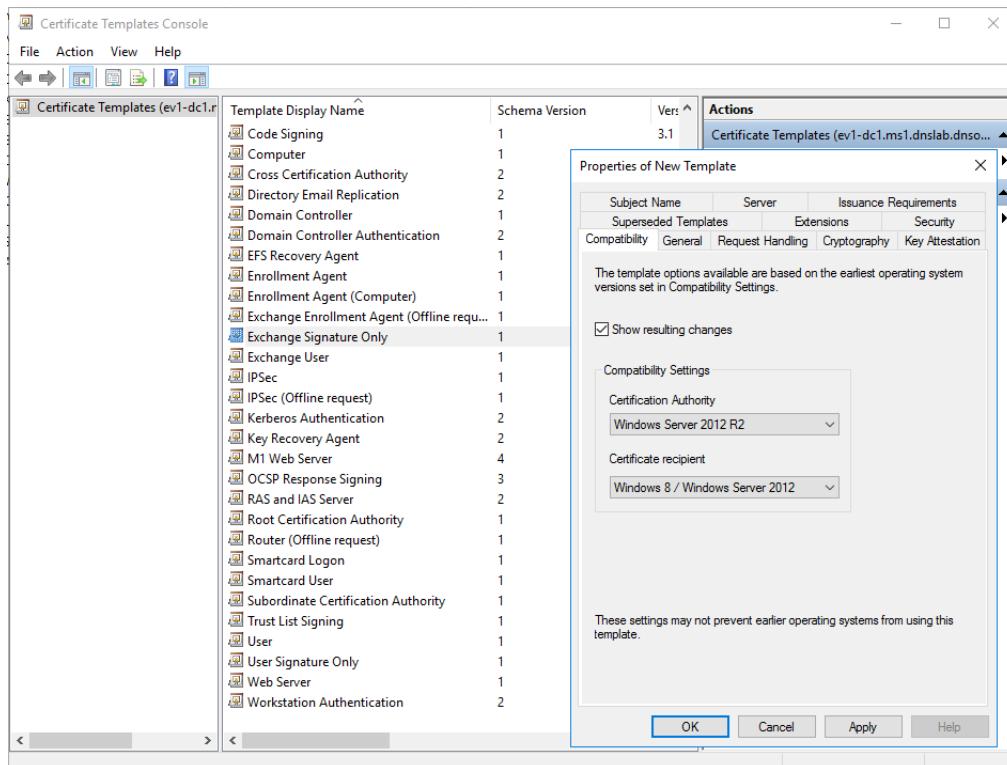
418 G.5.2 Issue S/MIME Certificates and Configure Outlook

419 To issue an S/MIME Digital Signature certificate to the user, go to the Issuing CA (ev1-issuingca).

420 1. Open the **Certification Authority** snap-in, right click on **Certificate Templates** and select
 421 **Manage**.

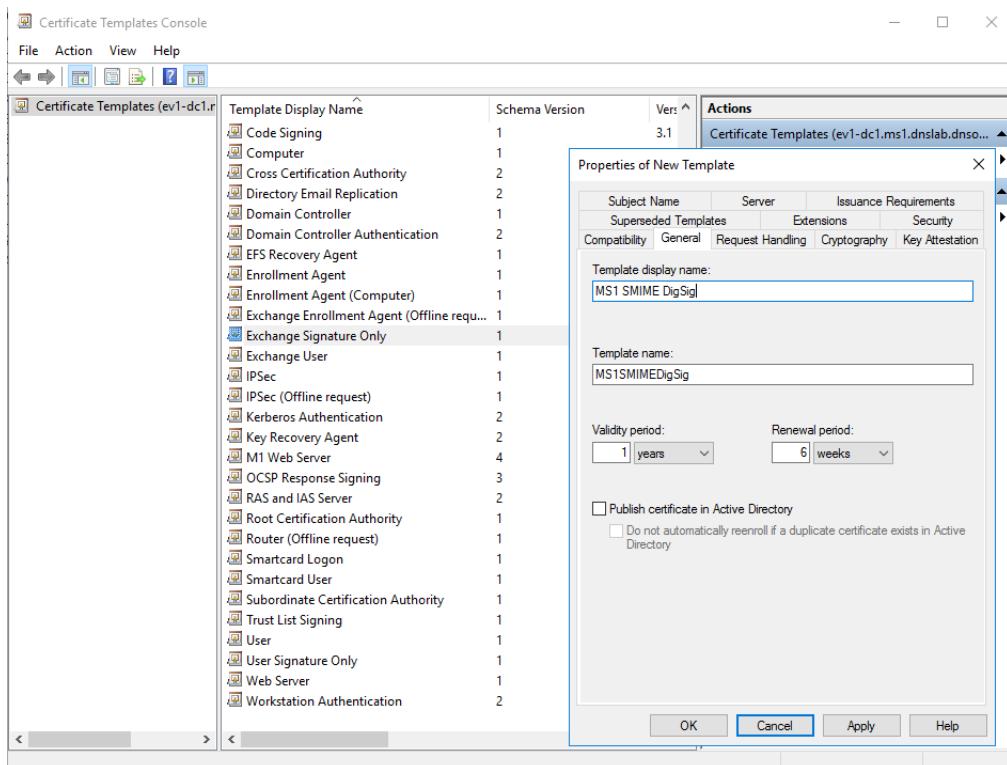
422 2. Find the **Exchange Signature Only** certificate template, right click and select **duplicate**.

423 3. Set **Compatibility** to **Windows Server 2012 R2**.



424

4. Within the **General** tab provide a name for the new template.

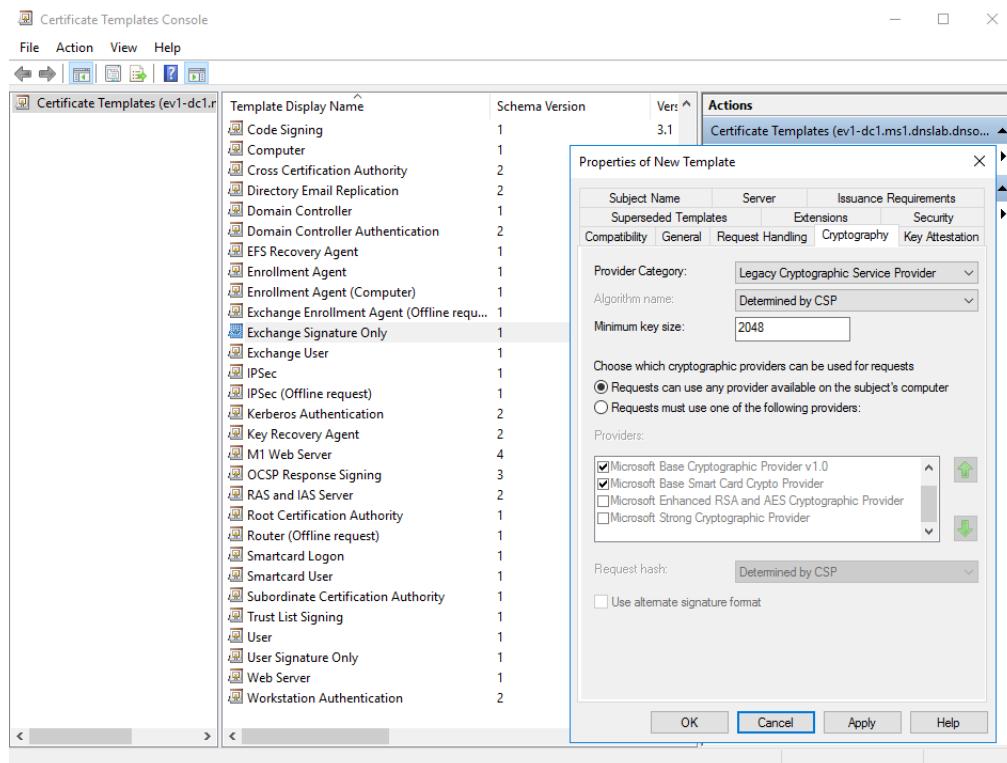


425

427

428

5. In the **Cryptography** tab select **Request can use any provider available on the subject's computer**.

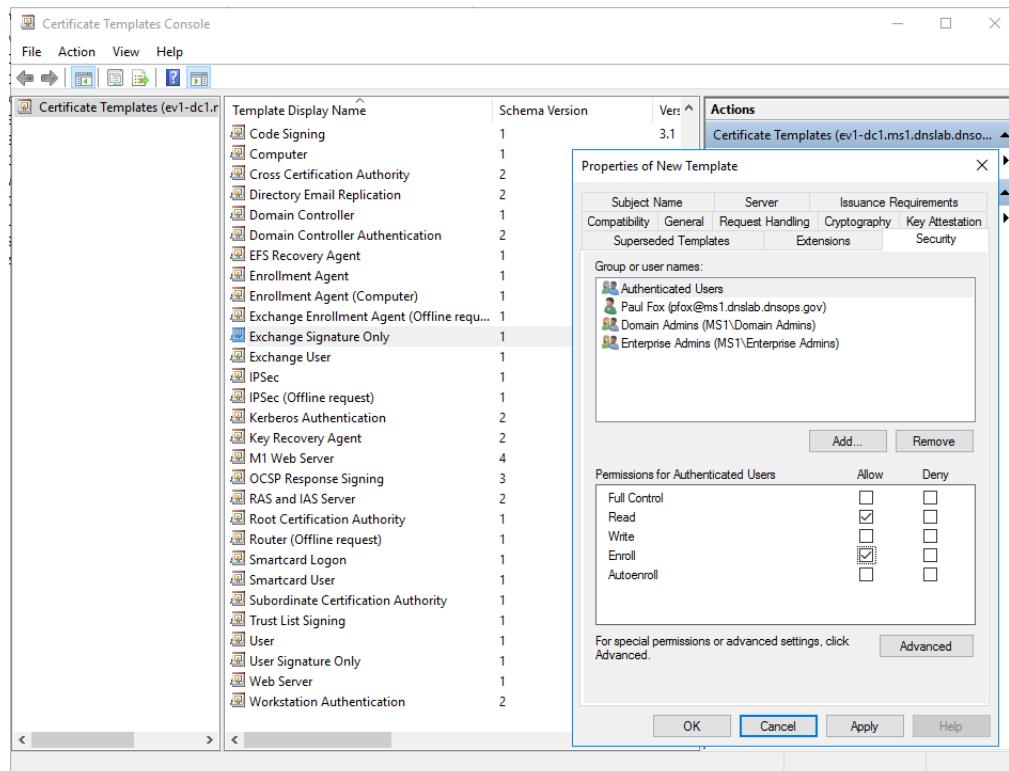


429

430

431

6. In the **Security** tab, select **Authenticated Users** from **Group or user names**, and allow **Read** and **Enroll**.



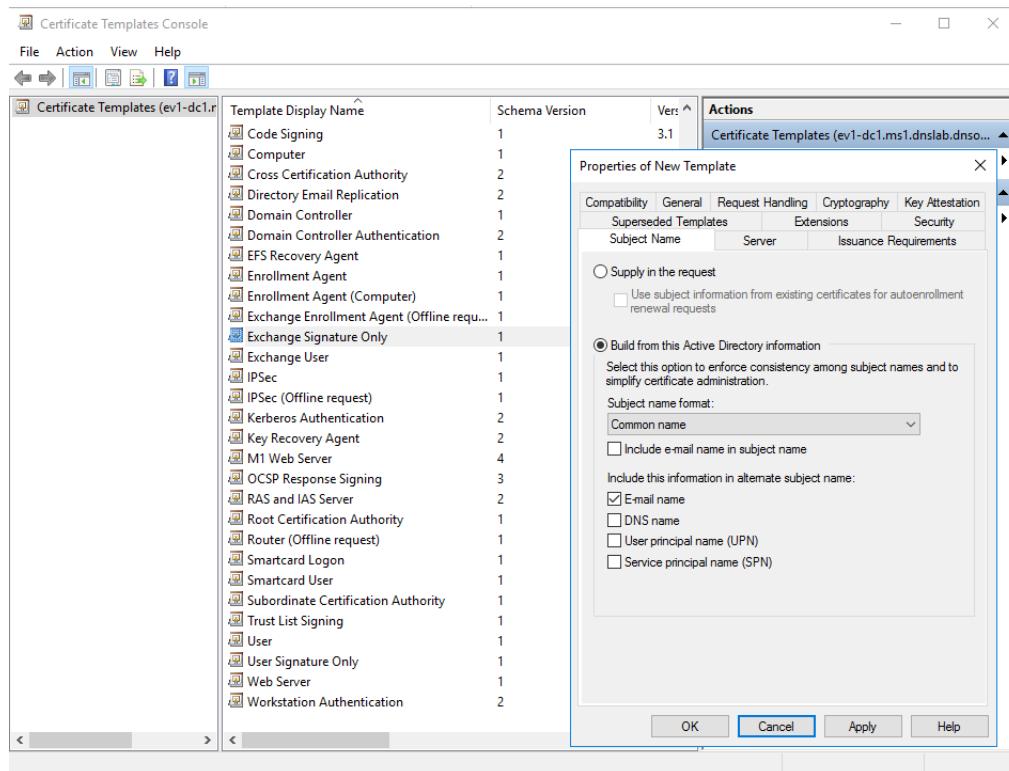
432

433

434

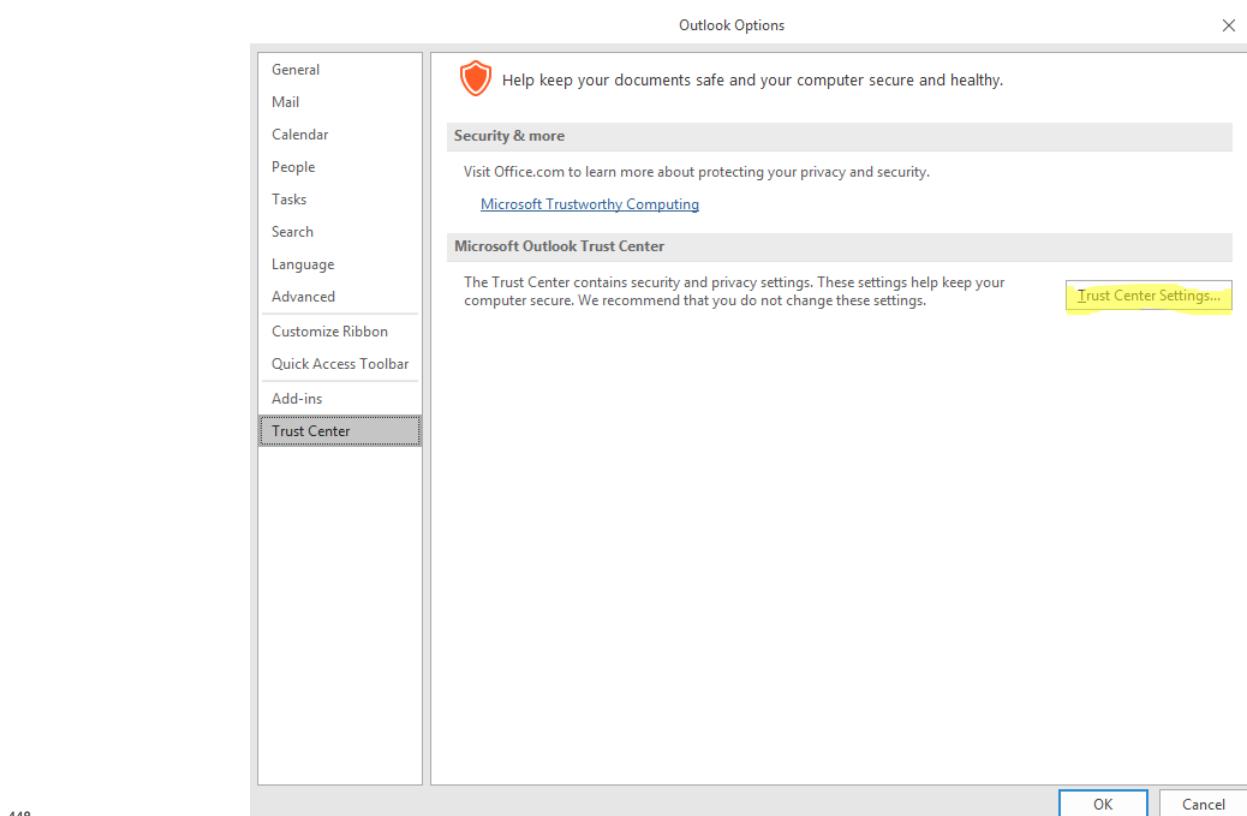
435

7. In the **Subject Name** tab, select **Build for this Active Directory information -> Email name** (note: make sure the mail attribute on the recipient's Active Directory object is populated with the correct email address)



436

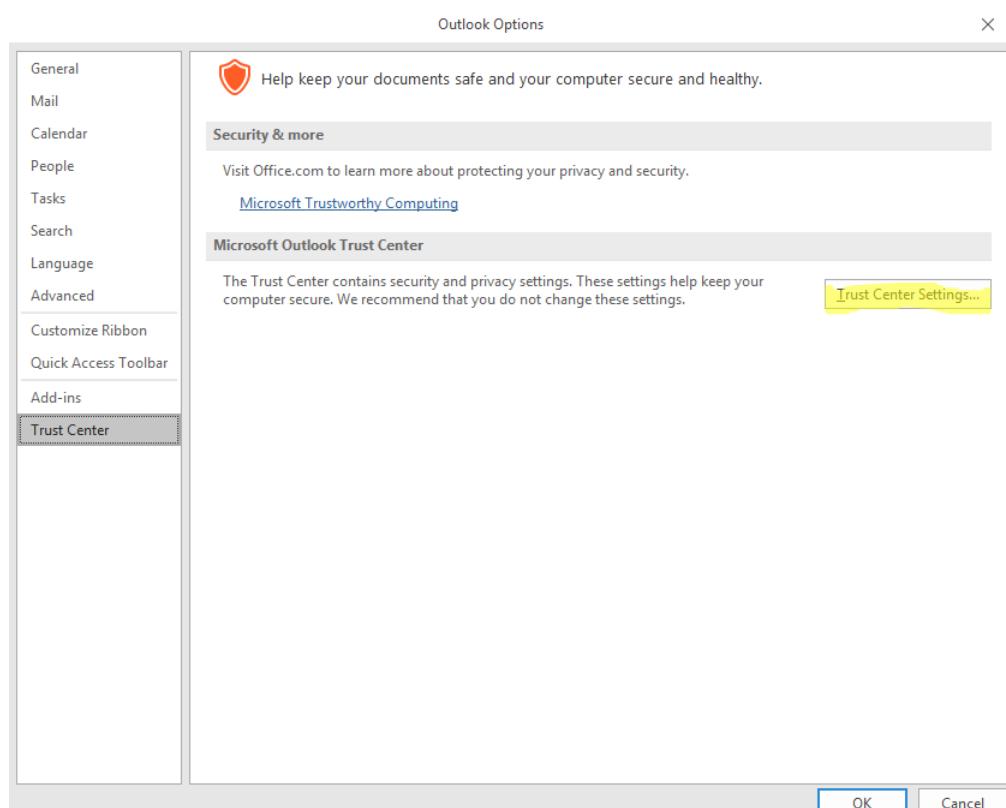
- 437 8. On the Windows 10 workstation, log on as the user that will receive the S/MIME Digital
- 438 Signature certificate. Start **certmgr.msc** -> **Personal** -> **right click: all tasks -> request new**
- 439 **certificate.**
- 440 9. Select the **Active Directory Enrollment Policy** -> select the certificate template that was just
- 441 created and follow the prompts.
- 442 10. Once completed, the S/MIME digital signature certificate will be in the user's Personal ->
- 443 Certificate store and can be used for S/MIME digital signature within Outlook.
- 444 11. To configure Outlook to use the new S/MIME certificate:
 - 445 a. Open **Outlook 2016**.
 - 446 b. Click on **File**, and then **Options**.
 - 447 c. In the left-hand menu click on **Trust Center**.
 - 448 d. Click on the **Trust Center Settings** box.



449

450

e. Click **Email Security** in the left-hand menu.



451

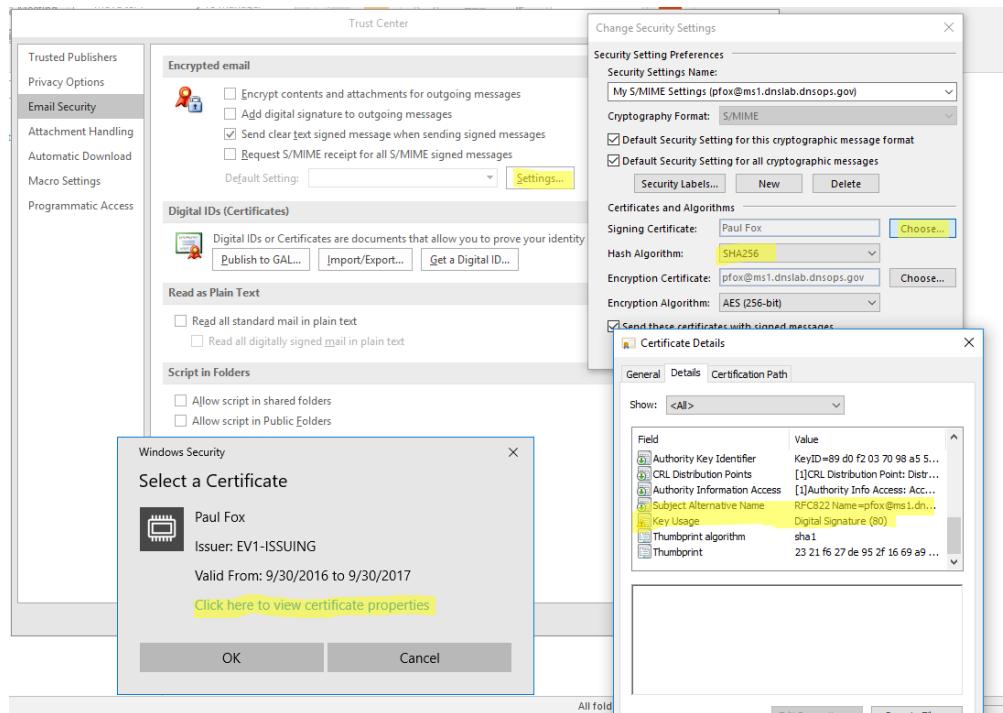
452 f. Click the **Settings** button within the **Encrypted Email** section.

453 g. Enter a name within the **Security Settings Name** field.

454 h. Select the **Signing Certificate** by clicking on the **Choose** button for the signing certificate, and select the **Hash Algorithm**.

455 i. If you have an S/MIME encryption certificate select the **Choose** button for the encryption certificate and select the **Encryption Algorithm**.

456 j. Select the radio button **Send these certificates with signed messages**.



459

Appendix H Installation and Configuration of DNS Authority, DNS Cache, and DNS Signer at the NCCoE

The NCCoE lab contained one DNS Signer appliance, and one VM instance each of DNS Authority and DNS Cache. These systems were not subject to special configurations beyond normal network configuration. The normal installation and setup for Secure64 products is found in the documentation (online at: <https://support.secure64.com/>).

There are no special configuration options needed for supporting DANE aware mail servers or clients with Secure64 DNS products. DANE Resource Record types are treated as any other valid DNS RRtype.

H.1 DNS Signer

Once the DNS Signer appliance is installed and initially set up, there are no special configuration options needed when deploying DANE to support email. Once a certificate is obtained (or generated) for the SMTP server, a TLSA RR needs to be generated and added to the zone. This can be done using one of the tools or websites described in Section 3.4 above. Once the TLSA RR is generated, the zone can be manually updated by editing the zone file or updated via dynamic update. Enterprises should follow any established procedure.

H.2 DNS Authority

Like DNS Signer, above, there is no difference between a standard setup of the authoritative server, and an authoritative server that hosts DANE RRtypes. Secure64 users should consult their product documentation on how to set up a DNS Authority instance.

H.3 DNS Cache

Like DNS Signer and DNS Authority, there are not additional steps in configuring a DNS Cache instance for supporting DANE. However, DANE requires the use of DNSSEC validation, so DNS Cache administrators (i.e. those that can enable the **cachdnsadmin** role) must enable DNSSEC validation and insure that the DNS Cache has a set of initial trust anchors.